

# **System of Systems (SoS) Architecture Centric Acquisition**

**SATURN 2010 Tutorial  
May 18, 2010**

**Mike Gagliardi  
John Bergey  
Bill Wood**



Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE <b>18 MAY 2010</b>		2. REPORT TYPE		3. DATES COVERED <b>00-00-2010 to 00-00-2010</b>	
4. TITLE AND SUBTITLE <b>System of Systems (SoS) Architecture Centric Acquisition</b>				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>Carnegie Mellon University ,Software Engineering Institute,Pittsburgh,PA,15213</b>				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release; distribution unlimited</b>					
13. SUPPLEMENTARY NOTES <b>SATURN 2010, May 17-21, 2010, Minneapolis, MN.</b>					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT <b>Same as Report (SAR)</b>	18. NUMBER OF PAGES <b>122</b>	19a. NAME OF RESPONSIBLE PERSON
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>			

# Tutorial Objectives

Present an outline with examples of the SoS Architecture Engagement

- Mission Thread Workshop, Architecture Challenge Workshop, SoS Architecture Evaluation, System and Software Architecture Evaluation, Acquisition Planning Workshop, etc.
- Capture comments and suggestions

Understand how the results of engagements can be applied within programs and organizations to improve program success

Demonstrate how different architectural based workshops can be used at different and various stages of SoS acquisition and development



# Introductions



# Outline

## Background

SoS Architecture Engagement Overview

Mission Thread Workshop

Legacy System and Software Architecture Evaluation

Architecture Challenge Workshop

SoS Architecture Evaluation

Architecture-Centric Acquisition



# Definitions - 1

A **System of Systems** is “a set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities.” [OSD Systems Engineering Guide for Systems of Systems, August 2008]

OSD SE Guide defines four types of SoSs:

- Directed
- Acknowledged
- Collaborative, Virtual



# Definitions - 2

An **Architecture** is the structure of components, their relationships, and the principles and guidelines governing their design evolution over time [IEEE Std 610.12 and DoDAF].

*An **SoS Architecture** is the structure of constituent systems, their relationships, and the principles and guidelines governing their design evolution over time.*

Need to elaborate on this to clarify.



# Elaboration

The structure(s) of the constituent systems include:

- Allocation of functionality to each constituent system
- End-to-end activity flows and communications, including operational, sustainment, development, and deployment activities.
- Externally visible properties and interfaces of the constituent systems, including behaviors, dependencies, use of shared resources, error handlings, etc.
- Relationship among organizational entities and the constituent systems at each phase of the SoS lifecycle.
- Rationale and governance policies, for example, criteria for decisions about constituent system inclusion, continued participation and termination.

Depending on the type of SoS:

- the point at which the structures are determined and by whom can vary
- the level of specificity and abstractions can vary





# Problem

Integration and operational problems arise due to inconsistencies, ambiguities, and omissions in addressing quality attributes between system and software architectures. This is further exacerbated in an SoS.

Example quality attributes: predictability in performance, availability/reliability, security, usability, testability, safety, interoperability, maintainability, force modularity, spectrum management

*Functionality and capability are important, but the architecture must be driven by the quality attributes. Identifying and addressing quality attributes early and evaluating the architecture to identify risks is key to success.*



# Common Symptoms

Failure to address quality attributes (non-functional requirements) in the architecture early will inevitably lead to symptoms such as these:

## Operational

- Communication bottlenecks under various load conditions throughout SoS
- Systems that hang up or crash; portions that need rebooting too often
- Difficulty synching up after periods of disconnect and resume operations
- Judgment by users that system is unusable for variety of reasons
- Database access sluggish and unpredictable

## Developmental

- Integration schedule blown, difficulty identifying root causes of problems
- Proliferation of patches and workarounds during integration and test
- Integration of new capabilities taking longer than expected
- Significant operational problems ensuing despite passage of integration and test
- Anticipated reuse benefits not being realized

**These symptoms often point to architectural deficiencies.**



# The Need for Augmented Mission Threads in DoD SoS Architecture Development

DoDAF is the SoS architecture framework for the DoD. It provides a good set of architectural views for an SoS architecture. However, it inadequately addresses cross-cutting quality attribute considerations.

System use cases focus on a functional slice of the system.

More than DoDAF and system use cases are needed to ensure that the SoS architecture satisfies its end-to-end functional requirements and quality attribute needs.

SoS end-to-end mission threads augmented with quality attribute considerations are needed to help develop, and later evaluate, the SoS and the constituent system/software architectures.



# Outline

Background

**SoS Architecture Engagement Overview**

Mission Thread Workshop

Legacy System and Software Architecture Evaluation

Architecture Challenge Workshop

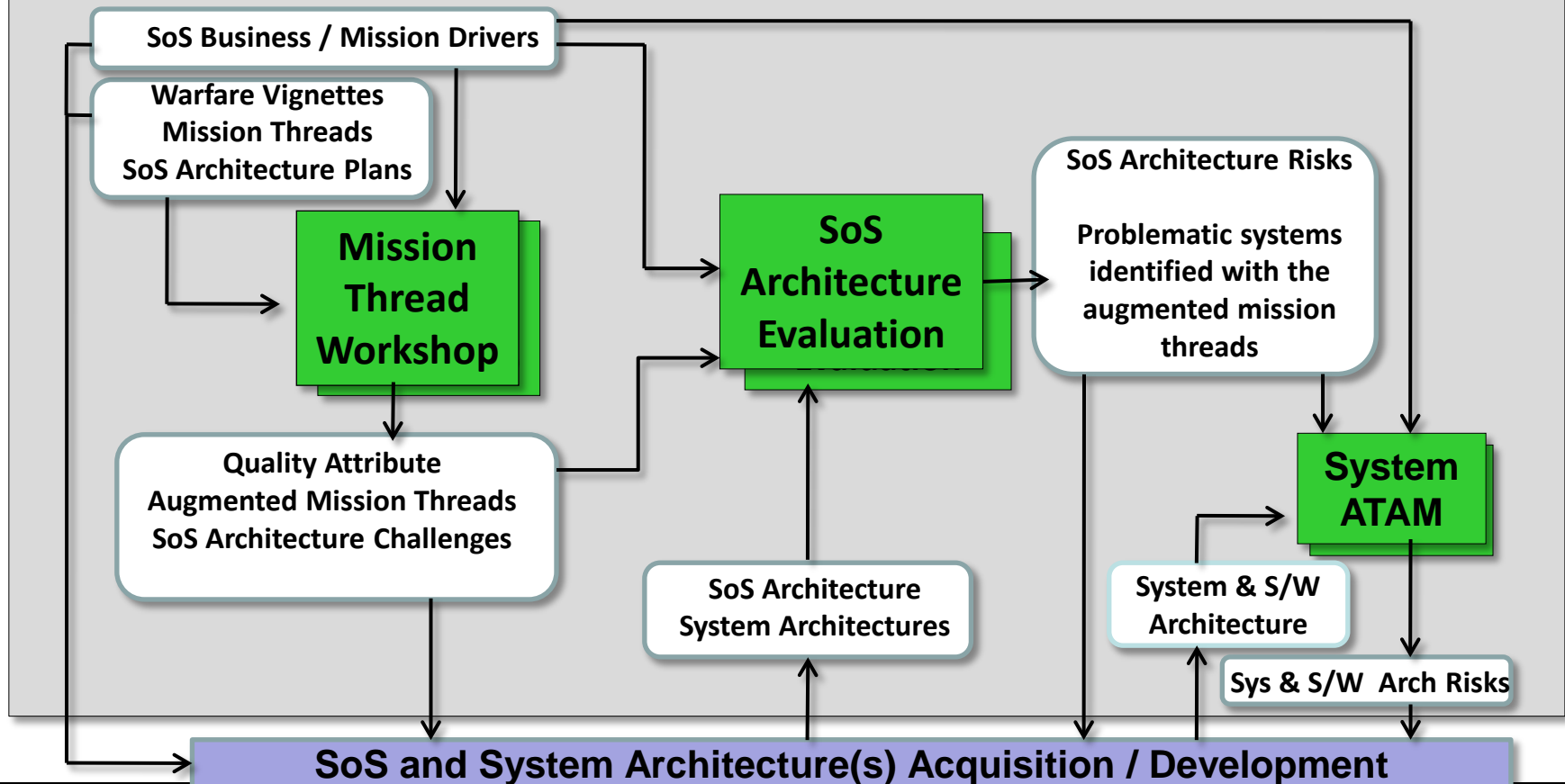
SoS Architecture Evaluation

Architecture-Centric Acquisition



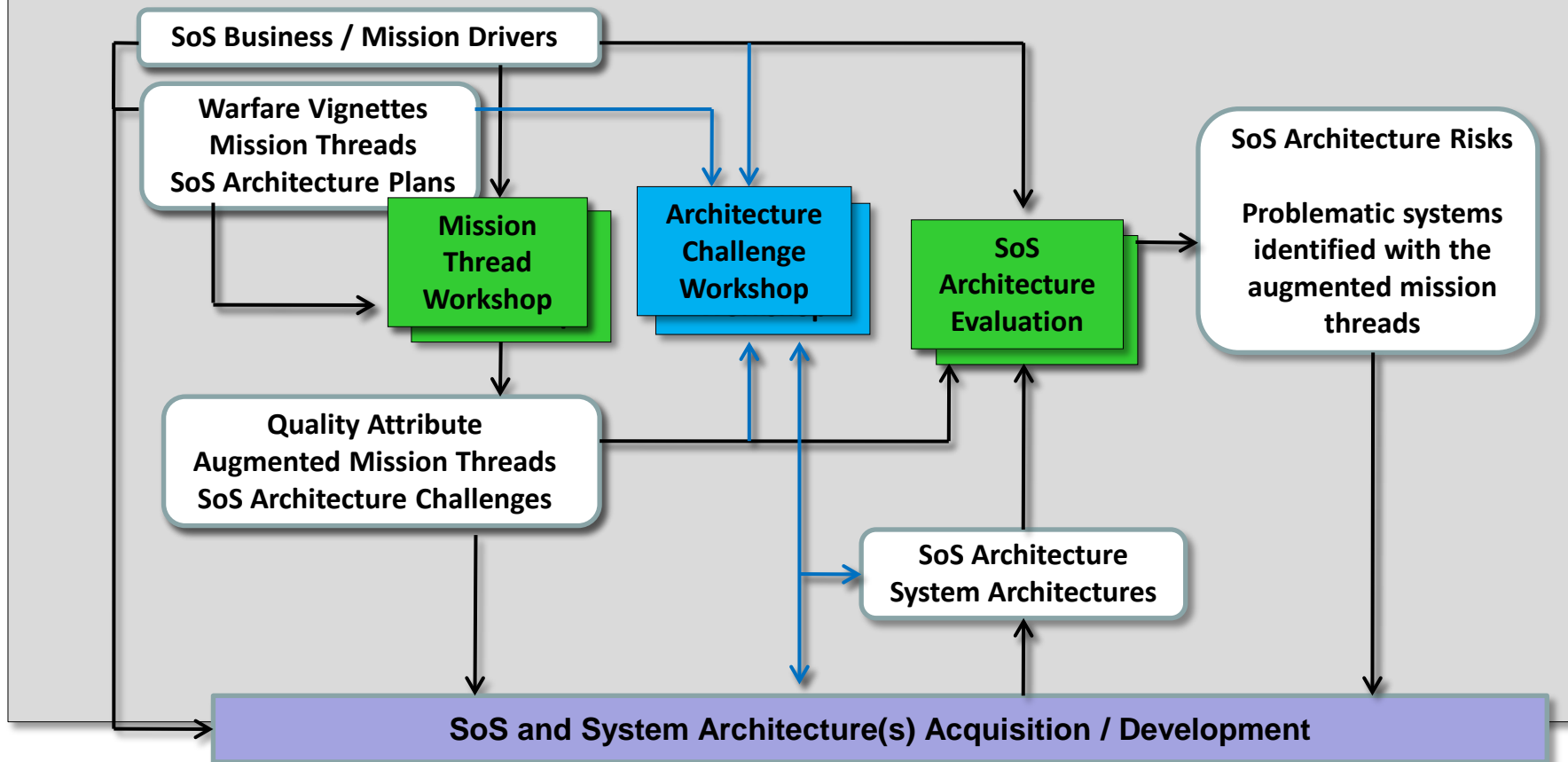
# SoS Architecture Quality Attribute Specification and Evaluation Approach

- Early elicitation of quality attribute considerations
- Early identification and addressing of architecture challenges
- Early identification and mitigation of architectural risks



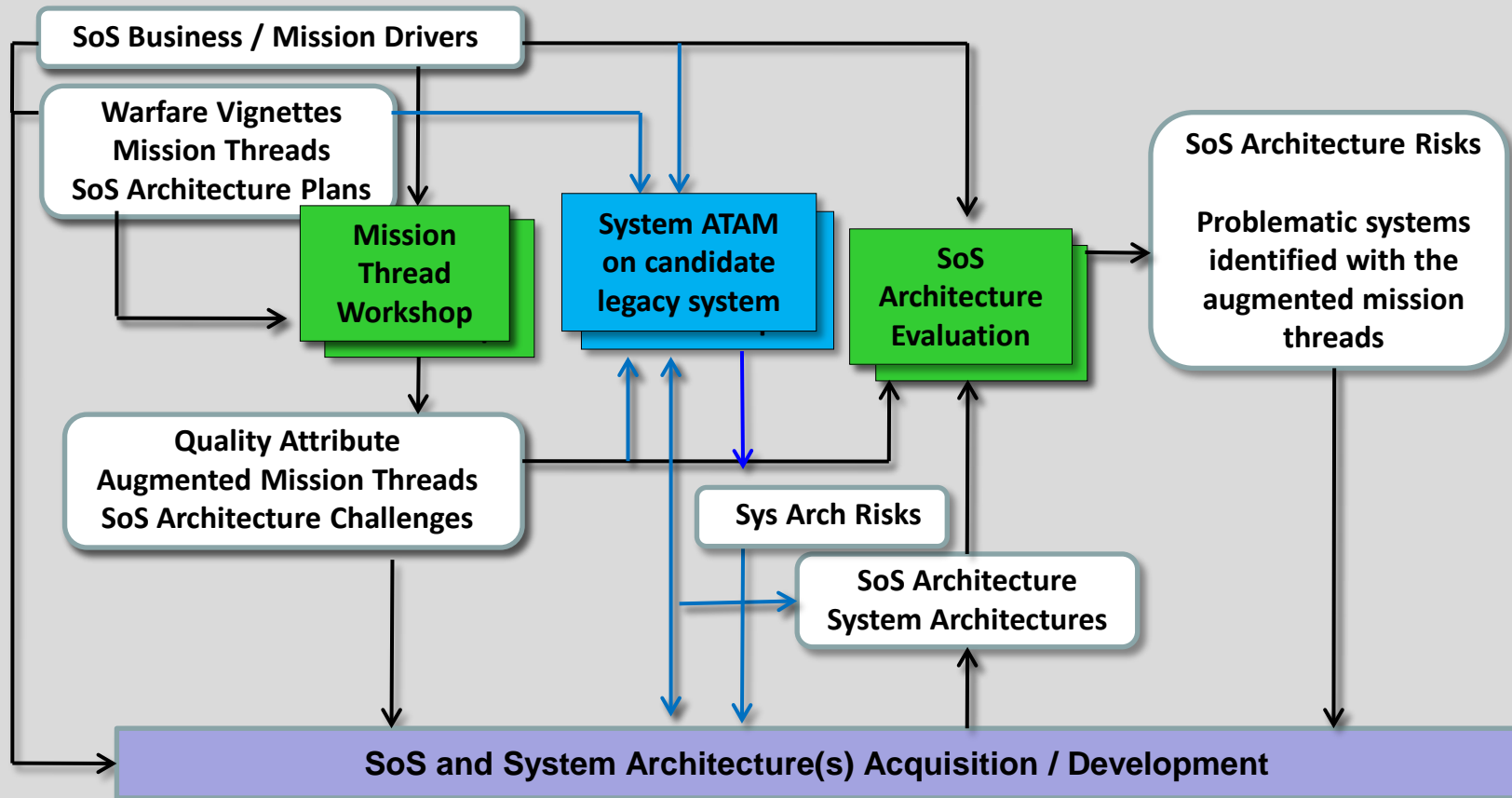
# Identify and Address Architectural Challenges - Early

- Early elicitation of quality attribute considerations
- **Early identification and addressing of architecture challenges**
- Early identification and mitigation of architectural risks



# Legacy System Architecture Evaluation - Early

- Early elicitation of quality attribute considerations
- Early identification and addressing of architecture challenges
- **Early identification and mitigation of architectural risks (e.g. candidate legacy system/software architecture evaluation)**



# Outline

Background

SoS Architecture Engagement Overview

**Mission Thread Workshop**

Legacy System and Software Architecture Evaluation

Architecture Challenge Workshop

SoS Architecture Evaluation

Architecture-Centric Acquisition





# Mission Thread Workshop (MTW) Purpose

The MTW ***augments SoS mission threads*** with quality attribute considerations that shape the SoS architecture and ***identifies SoS architectural challenges***, as early in the SoS development cycle as possible.

The mission thread augmentation is performed with inputs from key SoS stakeholders and is facilitated by the SEI.

The augmented mission threads and challenges are used to develop the SoS architecture and then later to evaluate the SoS and constituent system/software architectures.

There will be a series of MTWs depending on scope, scale, and schedule considerations.



# Definitions

**Warfare Vignette:** A description of the geography, own force structure and mission, strategies and tactics, the enemy forces and their attack strategies and tactics, including timing. There may be associated Measures of Performance (MOP) and Measures of Effectiveness (MOE). A vignette provides context for one or more *mission threads*.

**Mission Thread:** A sequence of end-to-end activities and events beginning with an opportunity to detect a threat or element that ought to be attacked and ending with a commander's assessment of damage after an attack. C4ISR for Future Naval Strike (Operational)

Sustainment: A sequence of activities and events which focus on installation, deployment, logistics and maintenance.

Development: A sequence of activities and events that focus on re-using or re-engineering legacy systems and new adding capabilities

Acquisition: A sequence of activities and events that focus on the acquisition of elements of an SoS, and the associated contracts and governance



# Vignettes Are the Starting Point – Example Wording

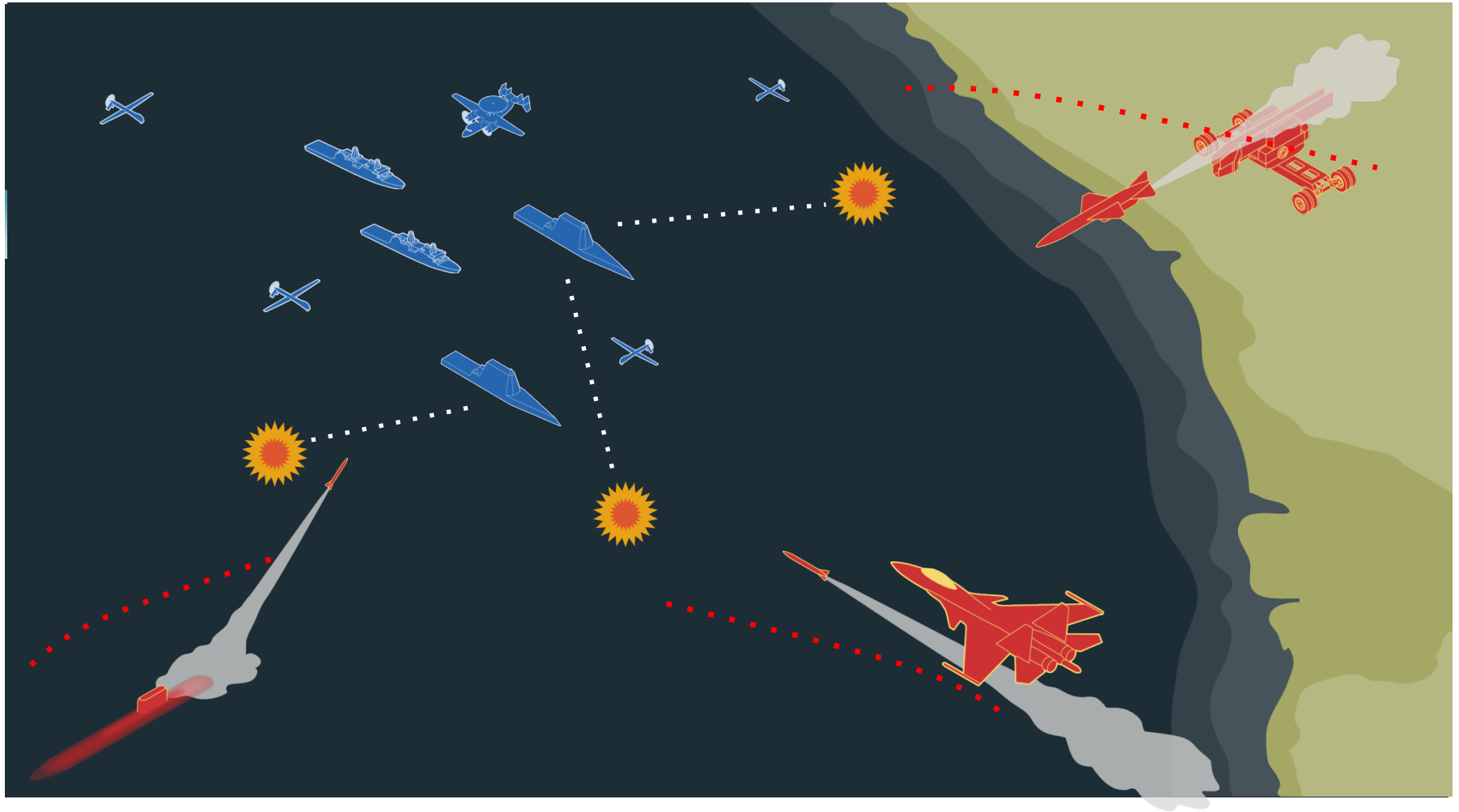
Two ships (Alpha and Beta) are assigned to integrated air and missile defense (IAMD) to protect a fleet containing two high-value assets (HVA). A surveillance aircraft SA and 4 UAVs are assigned to the fleet and controlled by the ships. Two UAVs flying as a constellation can provide fire-control quality tracks directly to the two ships. A three-pronged attack on the fleet occurs:

- 20 land-based ballistic missiles from the east
- 5 minutes later from 5 aircraft-launched missiles from the south
- 3 minutes later from 7 submarine-launched missiles from the west.

The fleet is protected with no battle damage.



# Vignettes Are the Starting Point – Example Context



# Mission Threads Flow from Vignettes – Example (Non-Augmented)

1. 20 land-based missiles launched - X minute window
2. Satellite detects missiles - cues CMDR
3. CMDR executes re-planning – reassigns Alpha and Beta
4. Satellite sends track/target data - before they cross horizon
5. Ships' radars are focused on horizon crossing points

...

N Engagement cycle is started on each ship

N+1. Aircraft are detected heading for fleet

N+2. SA detects missile launches – tells CMDR

N+3. CMDR does re-planning - UAVs are re-directed

N+4. FCQ tracks are developed from UAV inputs



# MTW Inputs - 1

## SoS Business and Mission Drivers Presentation (15 mins)

- A representative from the SoS stakeholder community presents the SoS business and/or mission drivers including the business/programmatic context, high-level functional requirements, high-level constraints, high-level quality attributes, acquisition strategy, etc.

## SoS Architecture Plans Presentation (30 mins)

- The SoS architect presents the architecture development plans including key business/programmatic requirements, key technical requirements and constraints that will drive architectural decisions, any relevant existing context diagrams, high-level SoS diagrams and descriptions, development spirals and integration schedule.



# MTW Inputs - 2

## Vignettes

- A description of the geography, own force structure and mission, strategies and tactics, the enemy forces and their attack strategies and tactics, including timing. There may be associated Measures of Performance (MOP) and Measures of Effectiveness (MOE).
  - An SoS will typically support multiple vignettes, i.e. multiple mission areas such as Air Defense, Ballistic Missile Defense, Replenishment, Mobility, etc.
  - Each vignette typically supports multiple mission threads

## Mission Threads, types:

- Operational - A sequence of activities and events beginning with an opportunity to detect a threat or element that ought to be attacked and ending with a commander's assessment of damage after an attack.
- Sustainment: A sequence of activities and events which focus on development, deployment and maintenance.



# Preparation

The SoS Program Manager develops a overview presentation on the SoS Mission / Business Drivers (see SoS Mission / Business Driver presentation template).

The SoS Architect develops an overview presentation on the SoS Architecture Plans (see SoS Architecture Plans presentation template).

The SEI meets with the SoS Architect and PM to:

- Determine if the vignettes and MTs are sufficient to proceed.
- Provide feedback on the presentations
- Reach agreement on scope and series of MTWs
- Identify Stakeholders
- Determine logistics





# Stakeholders are Key!

When developing the initial set of vignettes and MTs, it is critical to associate them with the key stakeholder types that will be necessary to participate in the Workshops.

There may be groups of stakeholder types that are not necessary for specific vignettes.

Example stakeholders: (leads in the following)

- Modeling and Simulations
- Integration and Test Facility (SIL)
- CONOPS, DRM, Operational Analysts,
- SoS, System and Software Architects
- Legacy System Architects



# Criteria for Development and Selection of Vignettes and Mission Threads

- Capability Coverage
- New requirements/capabilities
- Stressing the SoS and constituent systems
- New integrated existing capabilities

You can only do so many of these... make them count.



# Typical MTW Agenda

- 08:00 Welcome/Introductions/Opening Remarks (joint)
- 08:15 MTW Overview (SEI)
- 08:45 Business Drivers and Quality Attributes (Architect)
- 09:00 OV-1 & Vignettes Overview (Architect)
- 09:20 Break
- 09:35 Augmentation of 1<sup>st</sup> mission thread (SEI facilitated)
- 12:00 Lunch
- 13:00 Review OV-1 and vignette associated with 2<sup>nd</sup> mission thread (Architect)
- 13:20 Augmentation of 2<sup>nd</sup> mission thread (SEI facilitated)
- 15:00 Break
- 15:15 Review OV-1 and vignette associated with 3<sup>rd</sup> mission thread (Architect)
- 15:45 Augmentation of 3<sup>rd</sup> mission thread (SEI facilitated)



# Augmentation Process – Per Mission Thread

Two Passes over the Mission Thread:

- 1) For each event in the mission thread:
  - Elicit quality attribute considerations. Capturing any engineering issues, assumptions, challenges, additional use cases and mission threads (with QA context etc.)
  - Capture any capability and/or mission issues that arise.
- 2) For each Quality Attribute - elicit any over-arching quality attribute considerations
  - Capturing any over-arching assumptions, engineering issues, challenges, additional use case and mission threads (with QA context) etc.
  - Capture any capability and/or mission issues that arise.

Capture any MT extensions for later augmentation

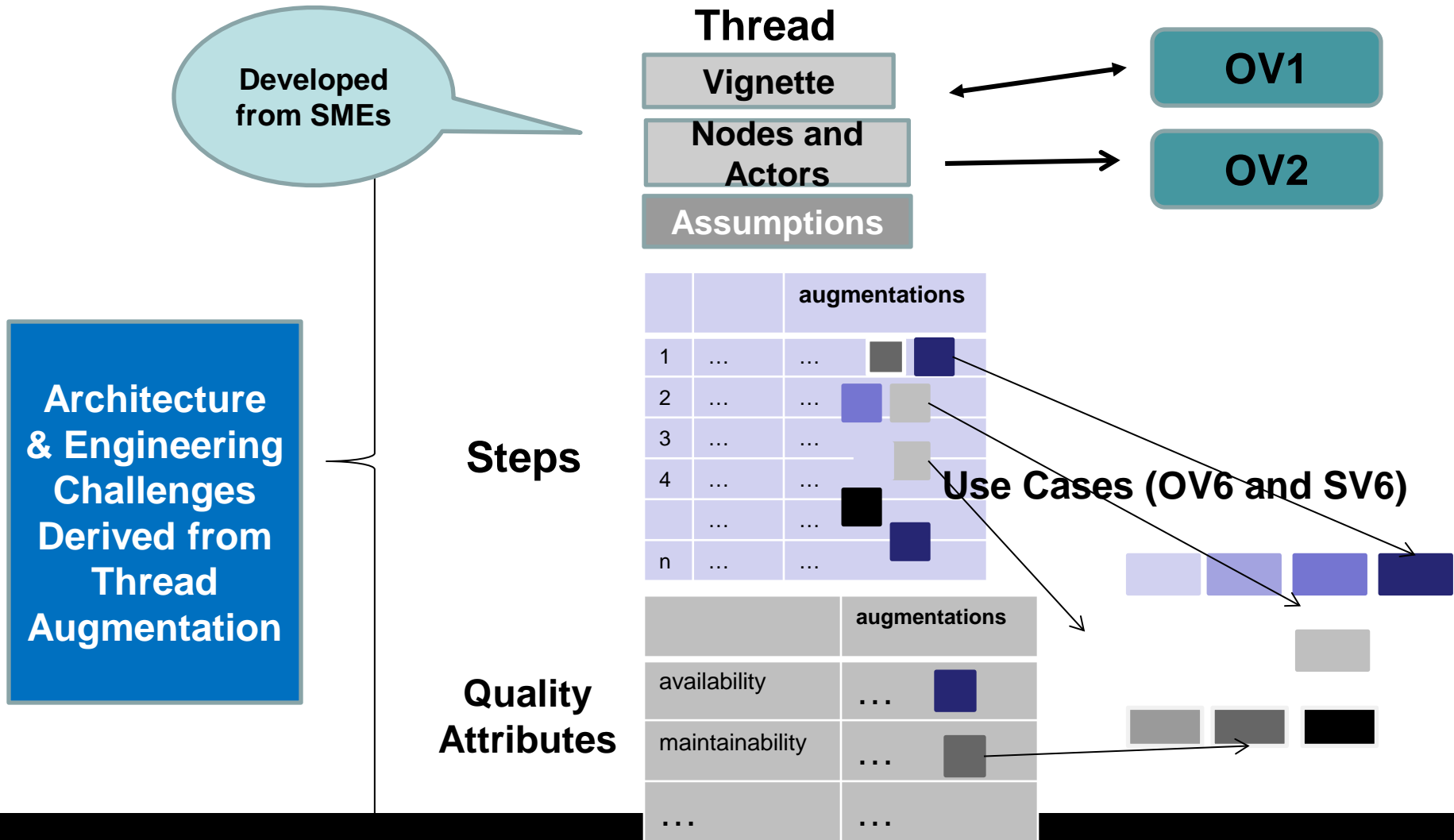
Capture Parking Lot issues – for organization, programmatic, non-technical issues that arise (will not be further pursued in the MTW).

**Stakeholder Inputs are Key.**



# Mission Thread

(augmented via the Mission Thread Workshop)



# Rules

SEI will provide the facilitation and scribing.

Side conversations, cell calls, etc. will not be allowed to disrupt the meeting.

Once an issue is identified and discussed, we will not allow it to be re-discussed. It will be noted at the appropriate place.

Will keep the discussions within scope.

Will not get into the details of potential solutions to issues.

Programmatic, organizational, and other non-technical issues will be noted, but not discussed in detail.



# Example MTW Walk-Through

At this point in the tutorial, we will switch to the MTW template which is partially filled in. We will walk through the MTW augmentation process using the DoD SoS example.



# Outputs

## Individual MTWs

- Augmented Mission Threads (.doc, using MTW template)
  - Over-arching quality attribute augmentations for the mission thread
  - Capability and mission augmentations to the mission thread
  - Quality attribute augmentations for each event in the mission thread
  - Identified mission/additional use cases (with context) and mission threads
- Challenges (briefing, vetted with sponsor)
  - Architectural, capability and mission challenges derived from the mission thread augmentations. Rolled up from the augmentation.
  - The MTW team will roll up challenges from the data and provide an out-brief of the challenges.
  - Any candidate legacy system architecture that may require architecture evaluation.

## SoS Architectural Challenges (briefing, vetted with sponsor)

- Report upon completion of series of MTWs:
  - SoS architectural challenges derived and rolled up from the mission thread augmentations; upon completion of the series of mission thread workshops for the SoS.





# Example Challenge: Resource Management

Description: The strategy for managing shared resources for the total ship needs further development. This is particularly important for radar (and weapons) resources in a missile defense context.

Impacts: Operational Availability, Mission Effectiveness, Multi-Mission Mode Capability

Recommendations: Work with Architecture IPT to address total ship hierarchical resource management strategy by providing radar, weapons, multi-missions and AoA inputs and evaluating strategy against resource needs.



# Examples of Rolled-up Challenges

End-End resource management strategy needs developed; esp. regarding issues dealing with supporting the number of missiles and radar coverage.

Fault model and recovery activities needs further definition and architectural guidance needs developed.

Degraded modes of operation strategy and associated architectural support needs developed.

Performance timelines and deadlines need defined and decomposed.

Manning/automation studies/analyses insufficient.

Sensor coordination between the two ships and the UAVs needs further analysis.



# Experiences with Mission Thread Types

## Over-Arching Mission Thread

- Creates a battlefield situation which exercises many capabilities from different nodes in the SoS and systems within some of the nodes
- Each step has a pointer to use cases for different systems and nodes

## Capability focused Mission Thread

- Create a mission thread showing how a warfighting capability (for example anti-submarine warfare) is conducted between a node and many external systems.

## New Doctrine Mission Thread

- Create a mission thread showing how the introduction of new capabilities (e.g. multiple UAVs giving FCQ tracks) can impact doctrine and provide better warfighting capabilities



# MTW – Initial Results - 1

The MTW and SoS Arch Evaluation methods adopted by a Navy and Army SoS programs and integrated into their architecture development process

Many of the identified challenges drove early risk mitigation activities (e.g. prototyping, EDM, white papers, modeling and simulation).

Many new use cases and additional mission threads identified. The QA considerations will be included in the use cases.

Excellent vehicle to promote communication between architects and stakeholders.

Capability and Mission Challenges were identified as well as Architectural Challenges.



# MTW – Initial Results - 2

SoS Architecture and Guidelines document is needed. Developed a template for use on Army and Navy SoS Programs.

Supports programs' DoDAF architecture development efforts.  
Normalized the OV-1s and informed and drove many subsequent DoDAF views (e.g. OV-5, OV-2, OV-3, OV-4, OV-6c, SV-5a, SV-4a, SV-1, SV-3)

3<sup>rd</sup> Party facilitation by the MTW facilitators enabled the leads to think about and participate in the discussions rather than trying to lead/control the meetings

Method worked for non-software elements, as well as software-intensive elements



# MTW Experiences – 1

Conducted a total of 35 MTWs (over 90 mission threads augmented),  
each MTW is a 1.5 day meeting

Plan 4 MTs per MTW, but expect to augment 3.

Expect 25-30 stakeholders to want to participate per MTW. Benefits from  
strong facilitation and independent 3<sup>rd</sup> party leadership.

Clients developed very good first pass vignettes and MTs after initial  
introduction.

Criteria for MT selection include: New capability, High perceived risk,  
proposal differentiators, etc.

DoDAF OV-1's were sufficient level of documentation going into the  
MTWs



# MTW Experiences - 2

Mission thread step elaboration focused on:

- Command authority, network communications, step constraints
- Manned vs Automated, timelines, planning considerations
- Availability and Survivability considerations
- Readiness, environmental conditions, start up/shut down
- New capabilities/extensions, don't be limited by current capabilities
- CONOPS considerations
- Assumption clarifications and issues

## Extensions

- Clients built some initially
- Added them as we go (to sideline discussions)



# MTW Experiences - 3

## Quality Attributes Considerations:

- Timeline decomposition often built into thread (weeks to seconds)
- Availability/ Degraded Operation / Resource Management under-developed
- Focus on operational MTs, separate MTW for development and support
- Over-arching MT pass collects much of the QA considerations
- Identified additional use cases and MTs (e.g. survivability)

## Challenges:

- Some challenges need to be kicked up to the SoS architecture level to address, while others need to be addressed by systems engineering
- Drives an SoS Architecture and Guidelines Document





# Outline

Background

SoS Architecture Engagement Overview

Mission Thread Workshop

**Legacy System and Software Architecture Evaluation**

Architecture Challenge Workshop

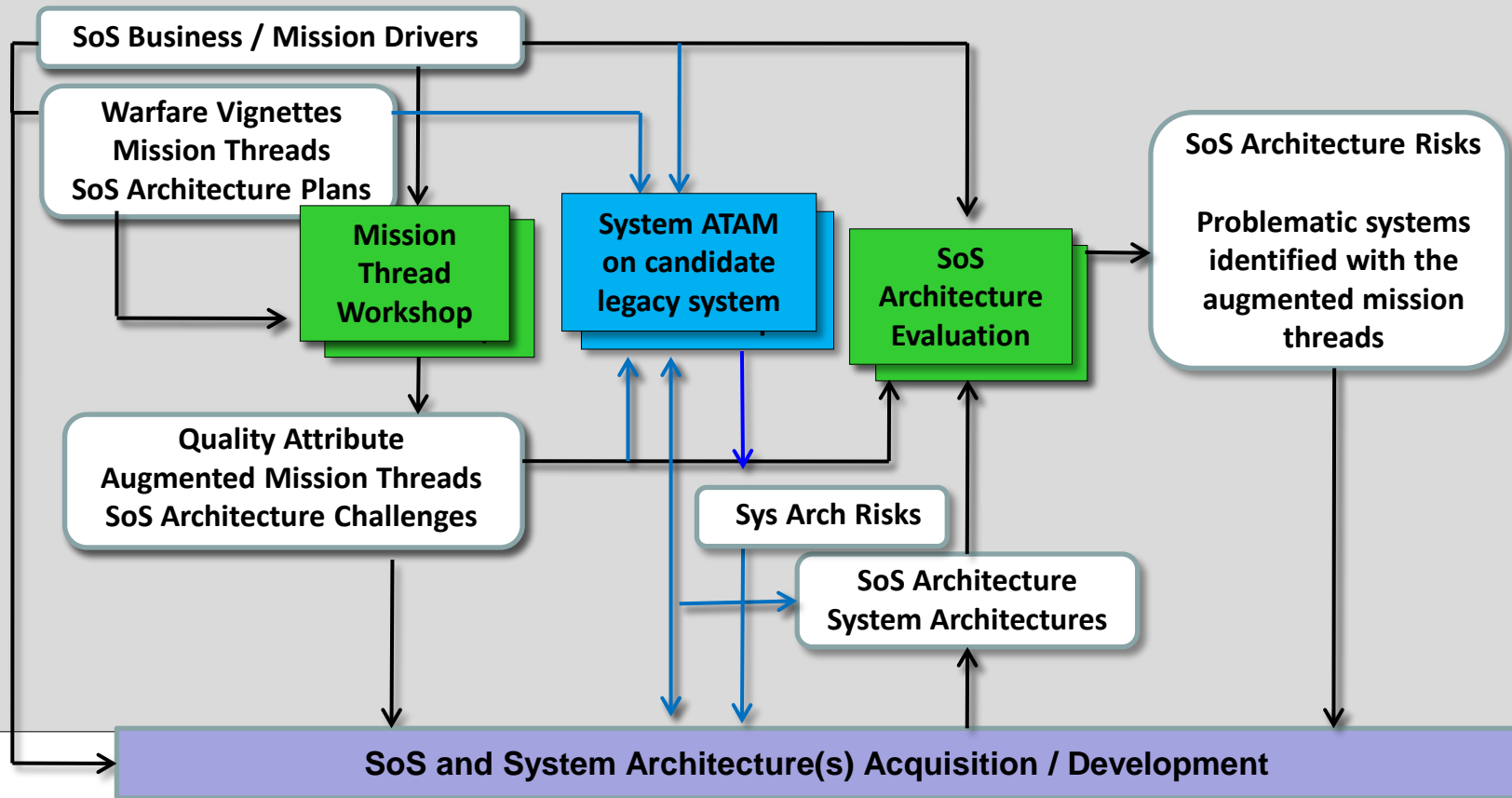
SoS Architecture Evaluation

Architecture-Centric Acquisition



# Legacy System Architecture Evaluation - Early

- Early elicitation of quality attribute considerations
- Early identification and addressing of architecture challenges
- **Early identification and mitigation of architectural risks (e.g. candidate legacy system/software architecture evaluation)**



# Purpose of the System ATAM

The System ATAM is a method that helps stakeholders ask the right questions to discover potentially problematic architectural decisions.

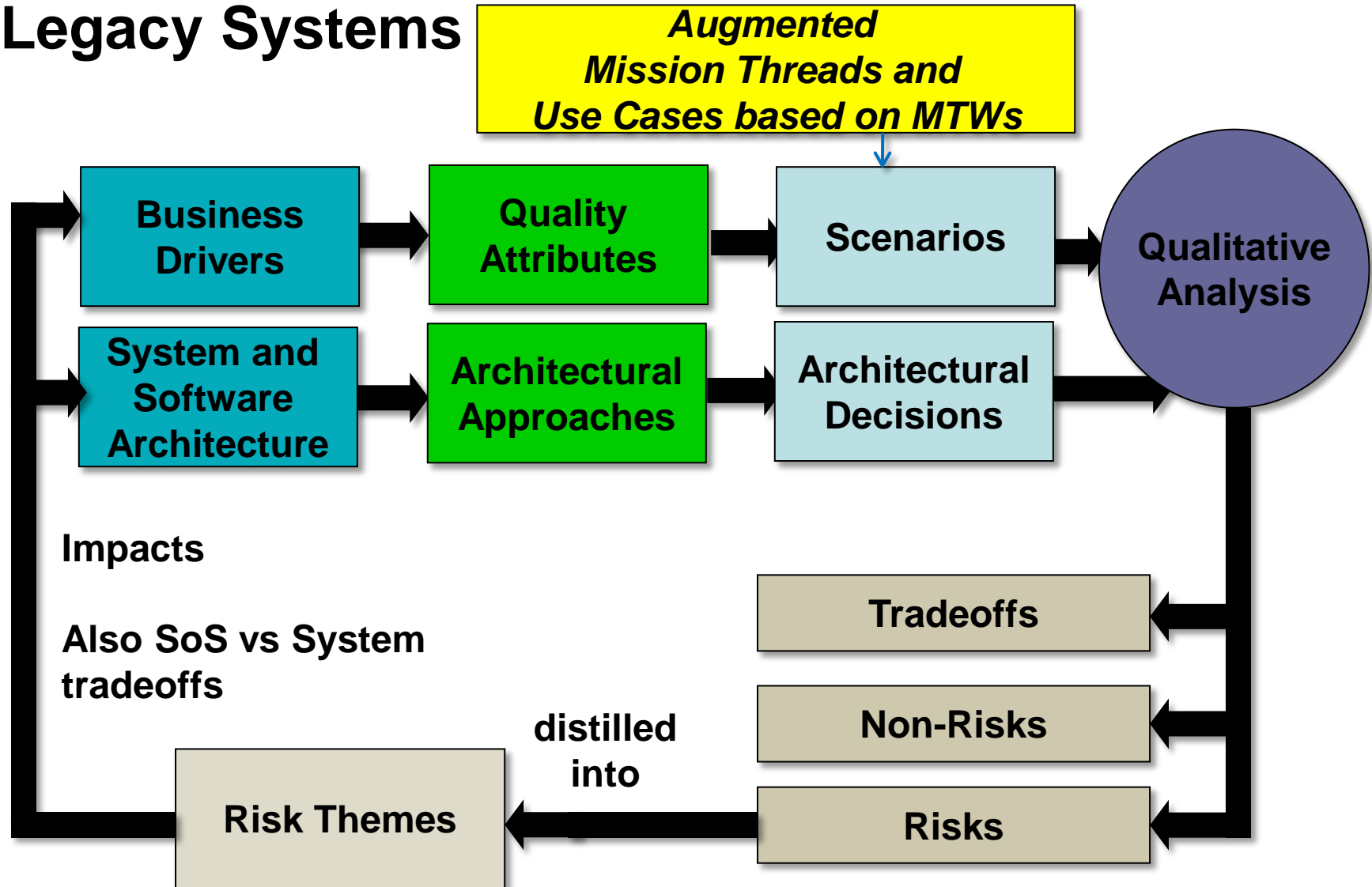
Purpose is to assess the consequences of **system and software architectural** decisions in light of quality attribute requirements and business goals; and to identify architectural risks.

The purpose is **NOT** to provide precise analyses; the purpose **IS** to discover risks created by architectural decisions.

Discovered risks can then be made the focus of mitigation activities. Tradeoffs can be explicitly identified and documented



# Conceptual Flow of System ATAM Variant for Legacy Systems



# Using the augmented mission threads to seed the system architecture evaluation

Issues from augmented mission thread identified in the MTW:

- The Defensive Engagement System may not be able to support the deconfliction timeline for 5 incoming missiles.
- The Defensive Engagement System may not have the capability to acknowledge Beta's acceptance of its assignment of 2 missiles.
- Is the Defensive Engagement System capable of sending track updates to the interceptor missiles that Beta had launched within the intercept timeline?

In preparation, the System ATAM lead meets with SoS and appropriate system architects to discuss what is in and out of scope concerning the system under analysis and if appropriate documentation exists.

Agreement is reached on the scenarios (based upon the augmented mission threads) with the understanding that additional scenarios can be added during the legacy system architecture evaluation.



# Examples of Scenarios

Scenarios address both system and software aspects. Consist of Stimulus, Environment and Response.

## Growth scenarios

*The Defensive Engagement System (DES) is able to support de-confliction of 7 incoming missiles using own-ship and external information within 5 seconds.*

*An upgraded DES is able to reduce the confliction time by 40% of 7 incoming missiles with no loss of existing functionality.*

## Exploratory scenario

*The DES is able to operate at up to 80% of its time budget for de-confliction of 7 incoming missiles with 8 coalition UAVs and 3 coalition helicopters operating in its vicinity.*



# Stakeholders and Evaluators

Stakeholders will consist of:

- System Architects of relevant, associated systems to system under evaluation
- SoS Architects who know the total system and how the system under evaluation is envisioned to fit in
- Relevant stakeholders of the system under evaluation in the areas of requirements, development, T&E, sustainment, M&S

ATAM evaluators will look to identify/expose potential system and software architecture risks, with the help of the stakeholders. Subject matter experts may be used on the evaluation team, if necessary.



# Walk-through of a scenario derived from augmented MT

*The Defensive Engagement System (DES) is able to support de-confliction of 7 incoming missiles using own-ship and external information within 5 seconds.*

- System architect identifies that currently DES can support 3 incoming missiles with 25% spare capacity within the latency bounds given the existing hardware.
- The software architect reveals that the system has a monolithic software architecture which is tightly coupled to the existing hardware.
- The architect identifies that upgraded hardware is available for the system which will provide the needed performance upgrade, but the software will need to be re-designed to take advantage of the upgrade.

SoS and DES architects and managers negotiate how to proceed based on architectural risks identified and associated risk mitigation options.





# Outline

Background

SoS Architecture Engagement Overview

Mission Thread Workshop

Legacy System and Software Architecture Evaluation

**Architecture Challenge Workshop**

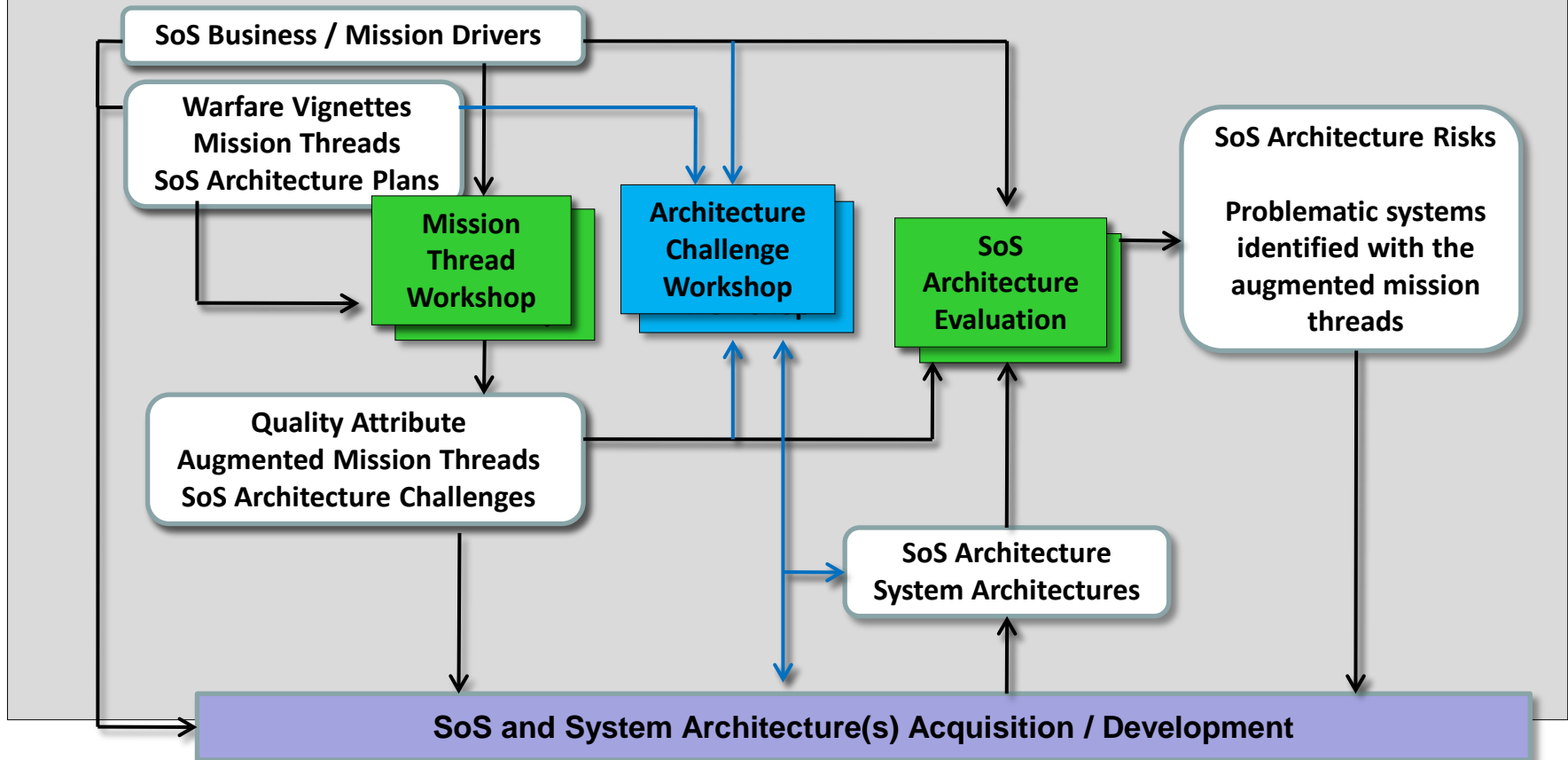
SoS Architecture Evaluation

Architecture-Centric Acquisition



# Identify and Address Architectural Challenges - Early

- Early elicitation of quality attribute considerations
- **Early identification and addressing of architecture challenges**
- Early identification and mitigation of architectural risks



# Purpose of ACW

Further analyze a challenge identified in the MTWs

- Typically 4 to 6 challenges are developed from MTWs
- MTW Principle stakeholders vetted the challenge

Develop a set of prioritized action items to resolve the challenge

- In a meeting with stakeholders

Assist the SoS architect to build CoAs for decision makers

- Rolling-up multiple ACW action-items



# What is a Challenge

The challenge slides were developed after MTWs

- Mapping challenges to vignettes/thread/step/assumption/QA
- Could be capability gap/architecture mismatch/ quality attribute problem

The challenge is broken into a number of aspects

- Typically 3 or 4

There are recommendations for each aspect

- Typically outlining an approach to resolving the aspect
- This has all been vetted with the principle stakeholders

**Unaddressed challenges will become risks**



# Example- Resource Management Challenge

## Challenge:

- The strategy for managing shared resources for the total ship which supports all the RM aspects across ship segments and missions needs further development. This is particularly important for radar (and weapons) resources in a missile defense context.

## Aspects:

- Reactive; Time Critical Planning; Mission/Voyage Planning; Degraded Operation

## Recommendations:

- Hold architectural challenge workshop to begin development of total ship resource management strategy. Work with Architecture IPT to address total ship hierarchical resource management strategy by providing radar, weapons, multi-missions and AoA inputs and evaluating strategy against resource needs.



# Outline of ACW

## Stages of ACW

- Preparation: preliminary technical analysis
- Conduct Workshop: with Stakeholders
- Follow-on: analysis and document



# Preparation

Perform a preliminary technical analysis of the challenge and build a briefing- jointly with client's principles

- Architectural elements involved
  - Nodes, systems, interfaces
  - Typically OV-1, OV-2, OV-3, SV-1, SV-2, SV-3, OV-4, OV-5
- Environmental and Other consideration
- Define Workshop scope

Build challenge-specific meeting templates for information capture.

Identify criteria for and select stakeholders to attend workshop.

Schedule the workshop.



# Workshop Scope

## Scoping Criteria

- What are the most critical new capabilities?
- To what nodes/systems do they apply?
- What challenge aspects are least understood and/or most stressful?
- What are the significant Quality Attributes?

## Scoping results example

- Focus on 2 (of 4) aspects, 2 (of 5) nodes and 4 (of 15) systems, 3 (of 9) operational capabilities, and 6 of (30) stakeholders
- Not all stakeholders need to address the challenge in a working meeting





# Stakeholder Identification

Don't invite all previous MTW attendees

Keep it small (~10)

Types:

- SoS Architect and PM
- System Architects and Sr Engineers
- Operations (if operational)
- SMEs – Challenge specific



# Conduct Workshop

Present the challenge briefing and preliminary templates with the stakeholders and update as necessary

- Each aspect of the challenge is treated separately
- Capture discussion points

Update the information based on discussion points

Review the “feasibility” of dealing with each aspect of the challenge

- legacy POR approach
- SoS studies already conducted
- What’s still unresolved

Action Items generated and prioritized

SEI contributes architectural experience and facilitates and scribes the workshop (based on challenge-specific templates)



# ACW Outputs for Each Aspect

## Summary of results of existing

- POR capabilities
- Results of studies, prototypes, experiments
- Gaps

## Identify

- Gaps in interactions between systems
- Potential approaches and associated risks
- Outline for architecture guidelines, further studies, etc

## Action items



# ACW Rollup

Review the action items created for each aspect

Combine/Upgrade them as necessary

Categorize them

- Importance, difficulty, time-frame

Assign priorities based on categorization



# ACW Follow-On

Identify impact on SoS Architecture Guidance And Precepts Document

Develop briefing to explain how the challenge is to be addressed, laying out the alternative courses of action (COA) for the decision makers.

- Include ROM and schedule for each COA
- Include risks mitigated and remaining with each COA

**Note: COAs may be developed from the results of multiple challenge workshops**



# Example - Resource Management Aspects

- **Reactive.** monitor resources, detect failure, issue alerts, and either recover from the failure or initiate degraded operation or transfer responsibilities
- **Time Critical Planning.** Services to support Threat based time-critical engagement planning.
- **Planning:** Mission and voyage planning (daily, port visits, UREP)
- **Degraded Operation:** Strategy for graceful degradation due to resource unavailability.

**Only interested in cross segment relationships**



# Example - Quality Attributes

This was developed from the MTW contributions to the MTW challenge table.

## Reactive:

- Availability, performance, automation, configurability, usability
- A single failure (electric generator) impacts multiple resources?

## Threat Plan:

- Performance, automation, configurability, usability
- Can we map a threat detection to a set of required resources?
- Can we re-assign resources to meet the threat?

## Daily Plan:

- Performance Predictability, configurability, usability
- Can we map timeframe of daily usage of resources to timeframe of assignment of capabilities?



# Example Aspect - Reactive

Loss of one of 3 electrical generators causes a reduction in capability in all segments

- Root Cause and Immediate automated actions
  - Many alerts will be generated based on circuit breaker openings
  - Some failure recovery activities will occur automatically
- Display capability loss to warfighter
  - 20% reduction in radar coverage
- Re-planning of resource usage and/or operations will be triggered in all segments
  - Immediate capability reduction- Short-term, mid-term, long-term
  - Casualties may be reported up the command chain
- Alternative COAs may be reported up the command chain and decisions fed down and implemented





# Example - Action Items

- Analyze the potential architectural approaches at the SoS architecture level for total ship resource management. Evaluate the constituent system/software architectural approaches supporting the total ship resource management strategy.
- Write a white paper outlining where root cause is resolved/unresolved for other significant failures
- Do a study to provide automated support to the Air Defense Commander to develop feasible COAs for multi-ship resolution when a threat will overload resources
- Determine how to re-engineer the POR resource manager to perform root cause analysis for a generator failure.



# Outline

Background

SoS Architecture Engagement Overview

Mission Thread Workshop

Legacy System and Software Architecture Evaluation

Architecture Challenge Workshop

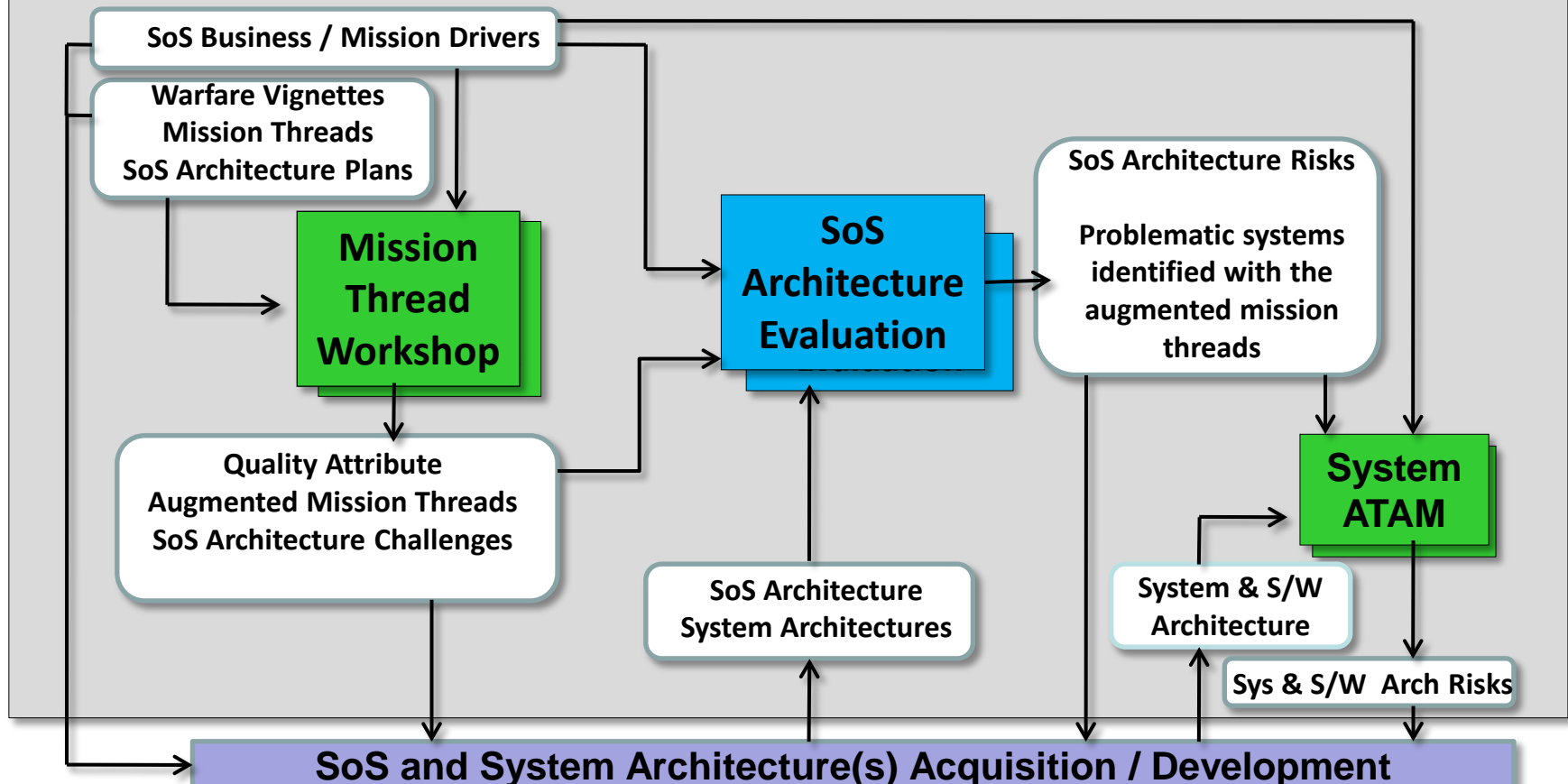
**SoS Architecture Evaluation**

Architecture-Centric Acquisition



# SoS Architecture Quality Attribute Specification and Evaluation Approach

- Early elicitation of quality attribute considerations
- Early identification and addressing of architecture challenges
- Early identification and mitigation of architectural risks



# SoS Architecture Evaluation Purpose

The SoS Architecture Evaluations ***identifies SoS architectural risks*** by probing the SoS architecture, using the augmented SoS mission threads and challenges, to evaluate the SoS architecture. It also ***identifies any problematic constituent systems that require further architecture evaluation.***

There will be a series of SoS Architecture Evaluations depending on scope, scale, and schedule considerations.



# Evaluation Approach - 1

## Similar to ATAM in some ways

- Appropriate architecture documentation required
- Stakeholders required throughout
- Architect(s) walk the augmented mission thread through the SoS architecture with evaluation team probing for risks, non-risks, etc.
- 2 day max per evaluation
  - Qualitative evaluation; not a precise, exhaustive evaluation
- Risks rolled up into risk themes
- Evaluation team required throughout
- Scoping is critical



# Evaluation Approach - 2

## Differs from ATAM in some ways

- Use existing augmented mission threads from the MTW
  - Requires execution of a MTW prior to evaluation
  - Mission threads augmentation not occurring during the evaluation
- Identify problematic areas for more focused architecture evaluation
- Initial preparation requires proper scoping and development of a scheduled series of SoS Arch Evals:
  - Ensure proper stakeholder representation; balance between not wasting anyone's time versus benefits of participation and communication.  
Depends on:
    - Mission thread “type” – operational, sustainment
    - Clustering of constituent systems per mission thread
  - Constrained by time it takes to go through a mission thread (1-2 per day)



# Evaluation Approach - 3

## Three stages

- Preparation
- Execution
- Roll-up and Follow-up



# Stage 1: Preparation - 1

Review results of MTW, noting the architectural challenges and expected resolutions; and highlight augmentations that require further explanation

Identify the mission threads for the SoS Arch Eval with the SoS architect

- Assume that only 1-2 mission threads can be evaluated per day max.

Develop and review the SoS business/mission drivers and the SoS and System/SW architecture presentations

Review SoS and system architecture documentation for sufficiency

Identify stakeholders (some to assist with the evaluation)





# Stage 1: Preparation - 2

Develop a schedule of the evaluations. Set up logistics and send out read-ahead with invitations

Walk-through one (partial) mission thread with architects for practice

Identify evaluation team

- Lead, Scribe, 3 Evaluators
  - ATAM evaluator qualified
- Domain SMEs (e.g. Communications, sensors, weapons, platforms, warfare experts)

Evaluation team reviews the inputs and becomes familiar with the SoS Architecture in advance of the evaluation



# Stage 2: Execution - 1

Note: 2 day max for each SoS Arch Eval

- probably will only get through 2-3 mission threads

Presentations:

- SoS Business/Mission Driver Presentation
- SoS Architecture Presentation
- Augmented Mission Threads for this evaluation
- Architectural Challenges from the MTW



# Stage 2: Execution - 2

## 1) Analysis for the Challenges

- SoS Architect describes how the architecture satisfies the specific challenge.

## 2) Analysis for the Mission Thread

- Walkthrough the architecture describing how the architecture satisfies the MT
  - Step by step probing capabilities and all highlighted QAs, looking for risks
  - Some hybrid of completing a step for all QAs and completing all steps for a QA.
- SoS Architect can hand over to system and s/w architects as needed

## 3) Analysis for the Over-Archiving Quality Attributes

- Final pass for any over-arching quality attributes that were not probed earlier.

Evaluation team probes for risks, identifies trade-offs, etc.

Always Identify business goals associated

Use “Parking Lot” for non-technical issues

Strong facilitation to stay on track; Do not go too deep in system architectures, whatever is architecturally significant at the SoS level.

## 4) Summarize findings in an out-brief



# Stage 3: Roll-up and Follow-up

At the end of each SoS Arch Eval:

- Output Briefing
  - SoS Architectural Risk Themes, Non risks, Trade-offs
  - Any non-architectural issues discovered
  - One example of an mission thread analysis with discovered SoS architectural risks, trade-off points and non-risks
  - Any problematic systems identified for future
  - Identify “parking lot” issues
- Summary Report of individual SoS Arch Eval
  - Detailed write-ups on the risk themes, non-risks, etc found during the evaluation
  - Summary of the SoS architecture, approaches, guidelines, etc
  - Summary of the SoS business and mission drivers, quality attributes, summarizing implications of any mismatches between SoS and systems



# SoS Arch Evals Roll-up

At the end of the series of SoS Arch Evals

- Evaluation team meets to roll-up the findings from the series of SoS Arch Evals
- Annotated Summary Briefing
  - SoS Architectural Risk Themes and Non-risks (rolled up)
  - Any non-architectural issues discovered (rolled up)
  - Identify problematic areas and schedule “focused” architecture evaluations (e.g. System & Software ATAM)
  - Recommendations
- SoS Arch Eval Summary Report
  - Detailed write-ups on the risk themes, non-risks, etc found during the evaluation
  - Summary of the SoS architecture, approaches, guidelines, etc
  - Summary of the SoS business and mission drivers, quality attributes, summarizing implications of any mismatches between SoS and systems
  - Recommended Next Steps



# Outline

Background

SoS Architecture Engagement Overview

Mission Thread Workshop

Legacy System and Software Architecture Evaluation

Architecture Challenge Workshop

SoS Architecture Evaluation

**Architecture-Centric Acquisition**



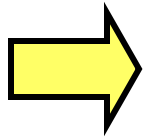
# Adopting an Architecture-Centric Acquisition Approach

John Bergey  
Michael Gagliardi

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213-2612



# DoD Acquirers and Software Architecture Practices



**What is** an architecture-centric acquisition approach?

**Why** should a DOD program adopt an architecture-centric acquisition approach?

**Where and when** is it appropriate in the DoD acquisition life cycle?

**How** can a DoD program adopt an architecture-centric approach and **what** does it involve?





# Architecture-Centric Acquisition

***Architecture-Centric Engineering*** is the discipline of effectively using architecture to guide system development.

[Klein 2009]

**Acquisition** is the process of *obtaining products and services through contracting*. Contracting includes purchasing, buying, commissioning, licensing, leasing, and procuring of designated supplies and services via a formal written agreement.

[Bergey and Fisher 1999]

**Architecture-Centric Acquisition** is the act of *using the architecture and architectural practices as a contractual means* to reduce risk and gain added confidence that the system being acquired will be capable of achieving its intended mission.

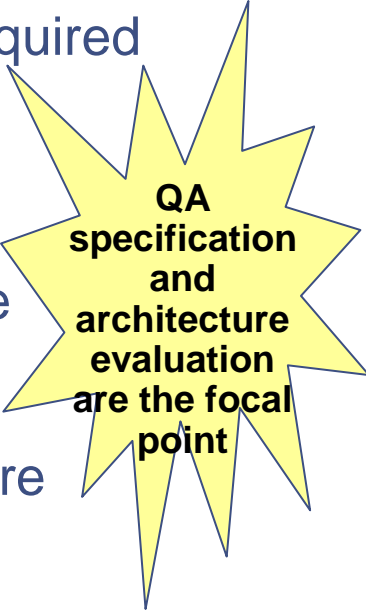
[Bergey 2010]



# Characteristics of an Architecture-Centric Acquisition

Architecture-centric acquisition involves

- Understanding the mission drivers for the system being acquired
- Understanding quality requirements of stakeholders
- Developing or selecting the software architecture
- Documenting and communicating the software architecture
- Analyzing and evaluating the software architecture
- Implementing the system based on the software architecture
- Ensuring that the implementation conforms to the software architecture
- Appropriately evolving the architecture over the system's life cycle
- Incorporating other architecture-related management and development activities to achieve specific program objectives

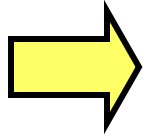


**QA  
specification  
and  
architecture  
evaluation  
are the focal  
point**



# DoD Acquirers and Software Architecture Practices

**What is** an architecture-centric acquisition approach?



**Why** should a DOD program adopt an architecture-centric acquisition approach?

**Where and when** is it appropriate in the DoD acquisition life cycle?

**How** can a DoD program adopt an architecture-centric approach and **what** does it involve?



# Why Is Architecture So Important? -1

Architecture is a common high-level communication vehicle for system stakeholders that is amenable to analysis and synthesis.

Architecture embodies the earliest set of design decisions about a system. These decisions

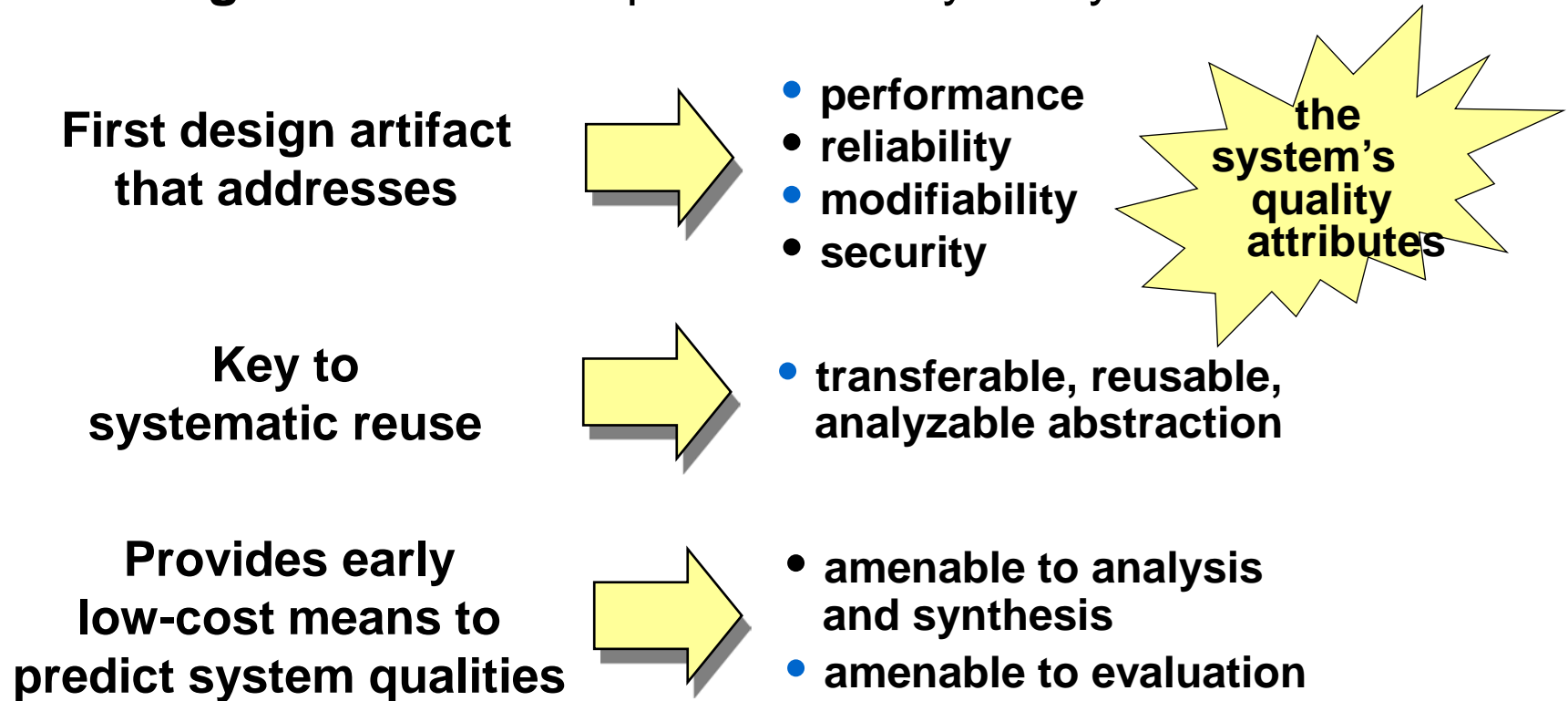
- are the most profound
- are the hardest to get right
- are most difficult to change
- ripple through the entire software development effort
- are most costly to fix downstream
- are critical to achieving mission/business goals

The earlier we reason about architecture tradeoffs, the better.



# Why Is Architecture So Important? -2

The **right architecture** paves the way for system **success**.



The **wrong architecture** usually spells some form of **disaster**.



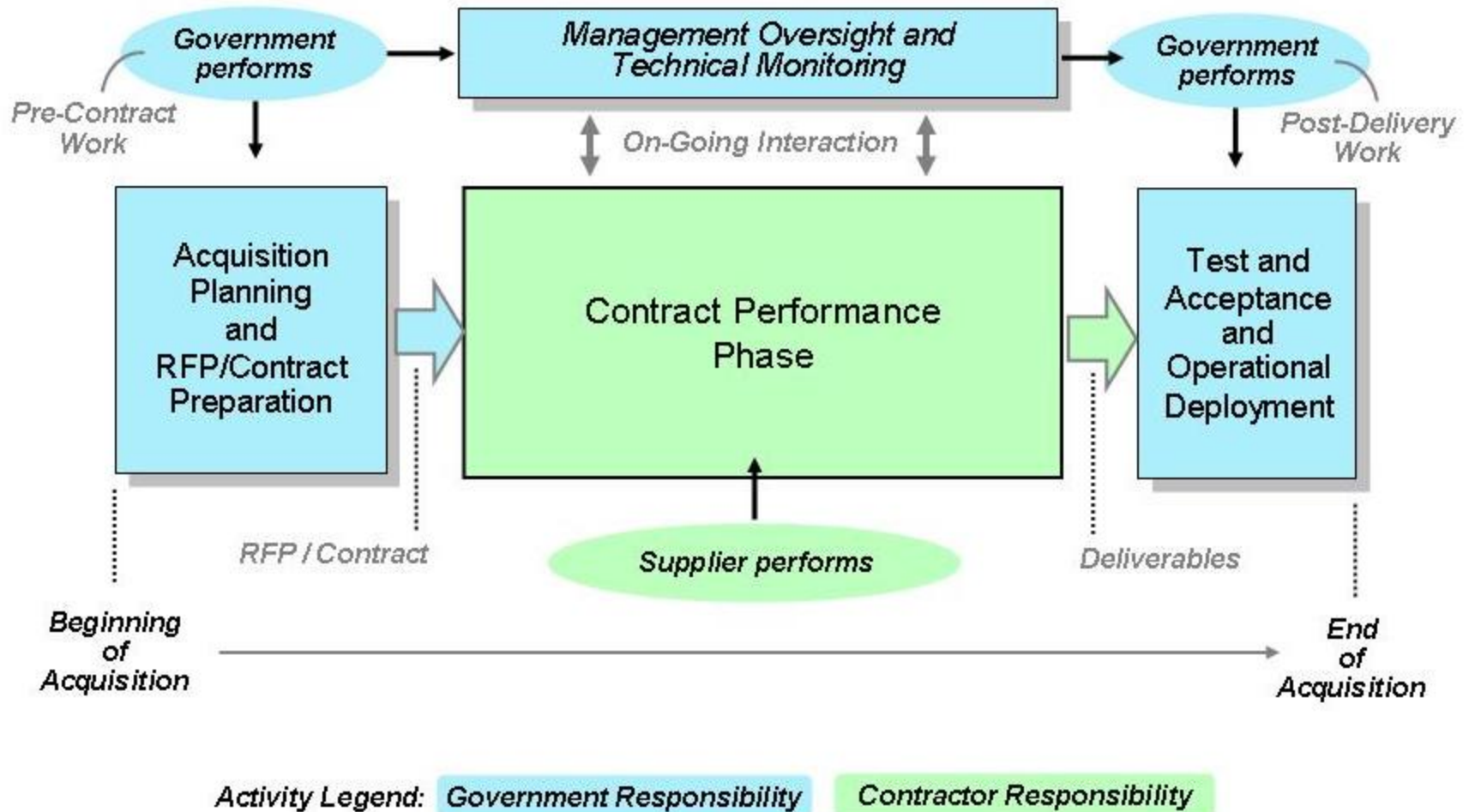
# Why an Architecture-Centric Acquisition Approach?

An architecture-centric acquisition approach

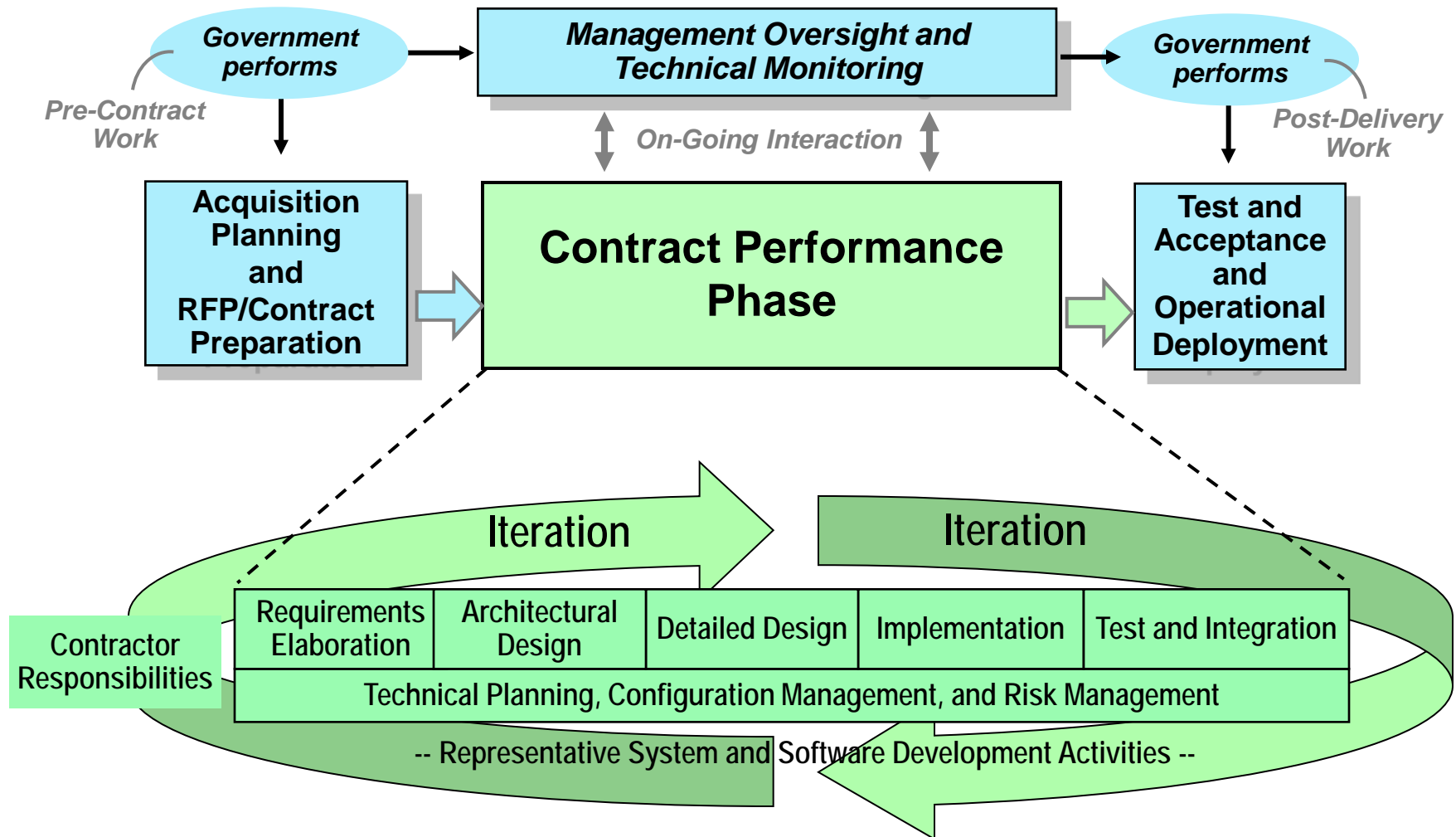
- enables a program office to perform its ***contract management and technical monitoring function*** with greater effectiveness
  - provides a focus that is commensurate with a program office's responsibilities, limited resources, time available, and key contractual events
  - provides early insight into critical requirements and design decisions that drive the entire development effort
  - enables software risks to be identified and mitigated early which results in fewer downstream problems and avoids costly rework
  - provides the knowledge base needed for cost-effective system evolution and sustainment
- results in the ***delivery of a more capable and higher quality product*** to the warfighter



# Responsibilities of an Acquisition Organization

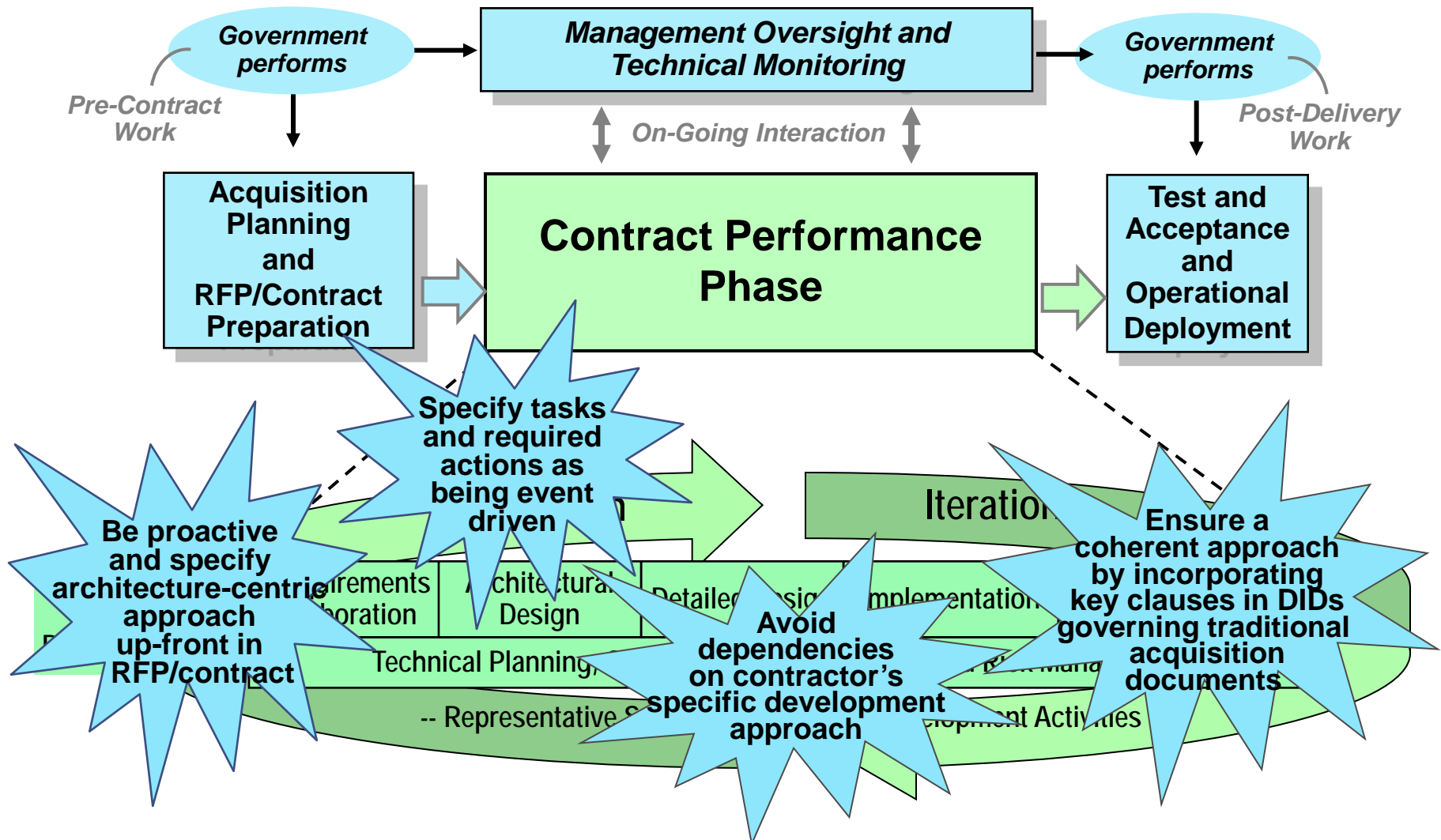


# Representation of Contract Performance Phase





# Representation of Contract Performance Phase



# Typical Scenario Describing Impact of Adopting an Architecture-Centric Acquisition Approach

**BEFORE:** There is no software architecture documentation.

**AFTER:** A software architecture description document is a contract deliverable.

**BEFORE:** The system's non-functional (i.e., quality) requirements that greatly impact the architecture design and software implementation are poorly defined.

**AFTER:** The system's quality requirements are specified in terms of a clear and concise set of quality attribute scenarios generated by key stakeholders.

**BEFORE:** The development contractor presents a couple of PowerPoint box-and-line drawings to describe the architecture and high-level software design.

**AFTER:** The software architecture description includes a comprehensive set of views (e.g., module decomposition, allocation, run-time) that is amenable to analysis

**BEFORE:** The proposed software design is not appropriately analyzed or evaluated.

**AFTER:** The software architecture is evaluated with stakeholder participation and risks (and risk themes) are identified and appropriately documented.

**BEFORE:** Architecture development is ad hoc and not based on careful analysis.

**AFTER:** As a result of the architecture evaluation, the development contractor creates a risk mitigation plan and presents it at the Preliminary Design Review (PDR).

**BEFORE:** Plans for architecture evolution are ad hoc and not based on careful analysis.

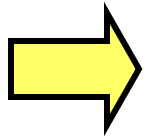
**AFTER:** In conjunction with the risk mitigation plan the development contractor develops a software architecture improvement roadmap based on an incremental software development approach.



# DoD Acquirers and Software Architecture Practices

**What is** an architecture-centric acquisition approach?

**Why** should a DOD program adopt an architecture-centric acquisition approach?

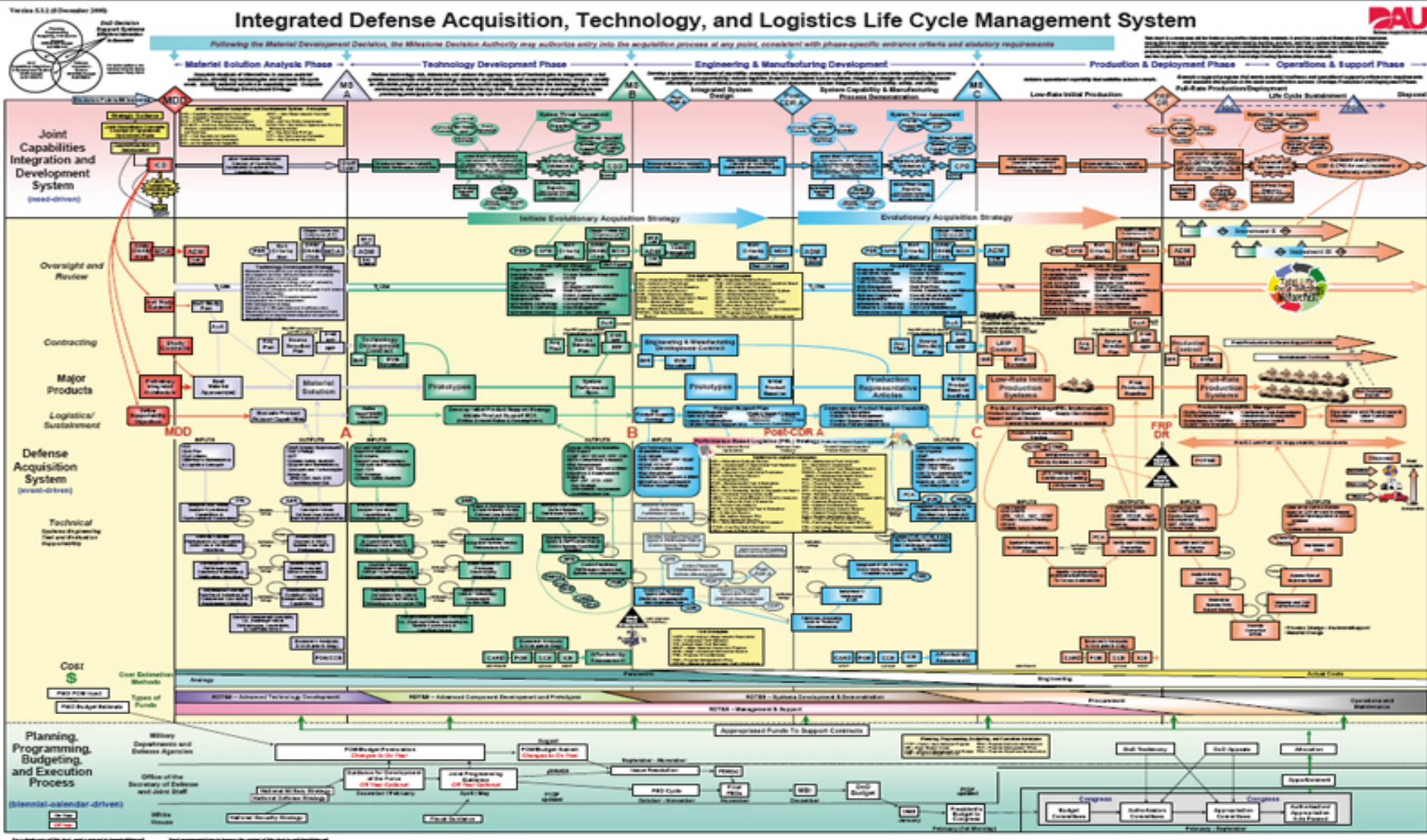


**Where and when** is it appropriate in the DoD acquisition life cycle?

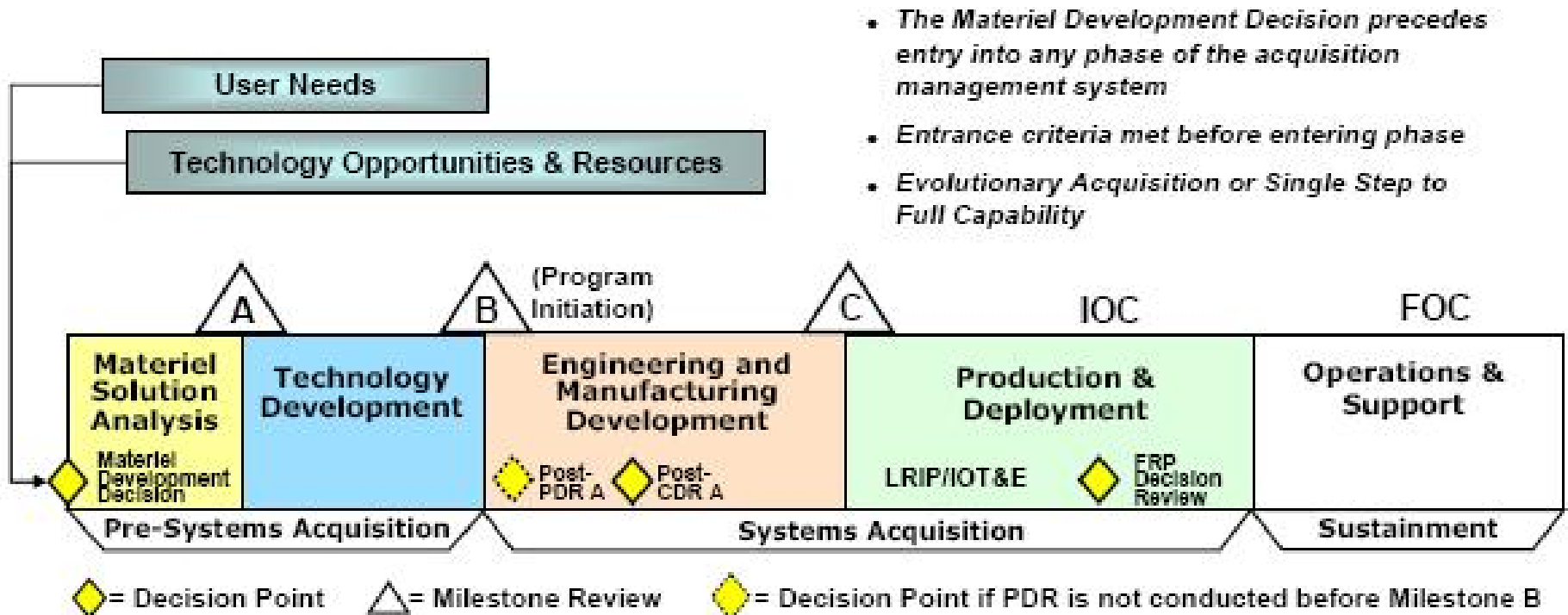
**How** can a DoD program adopt an architecture-centric approach and **what** does it involve?



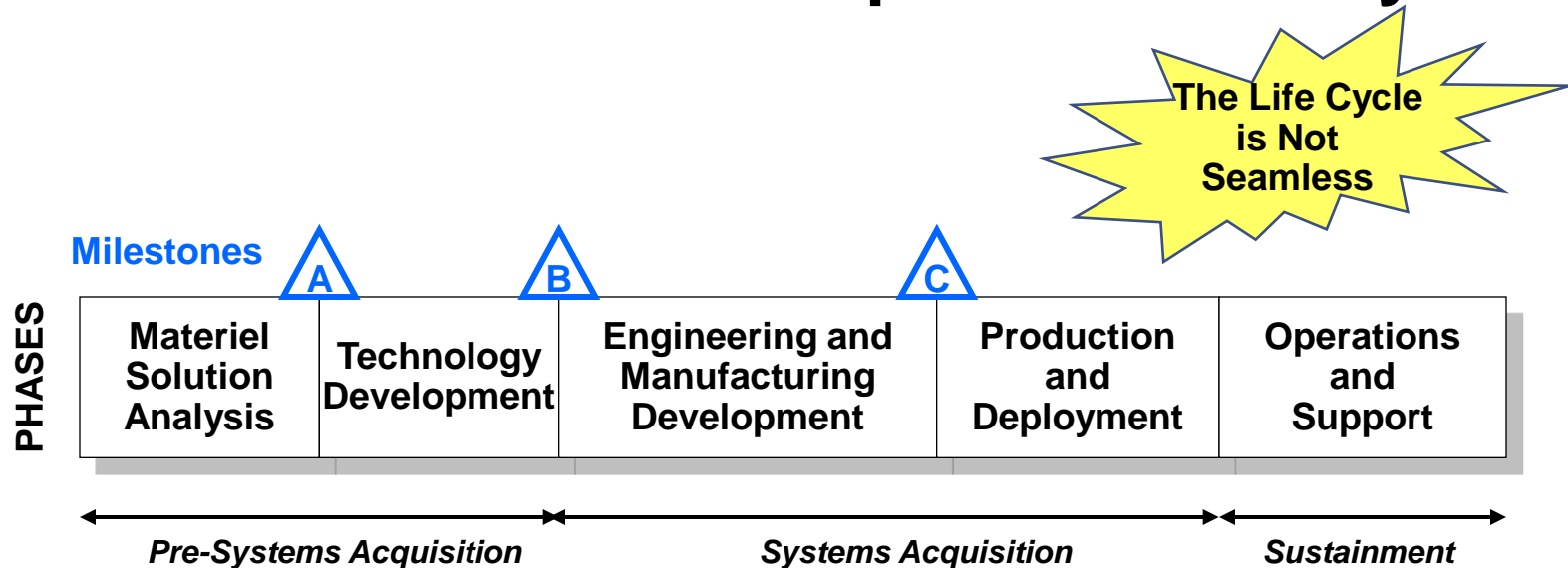
# “Big Picture” View of DoD Acquisition Life Cycle



# New DoD 5002 Acquisition Life Cycle



# Overview of DoD Acquisition Life Cycle



## Definition:

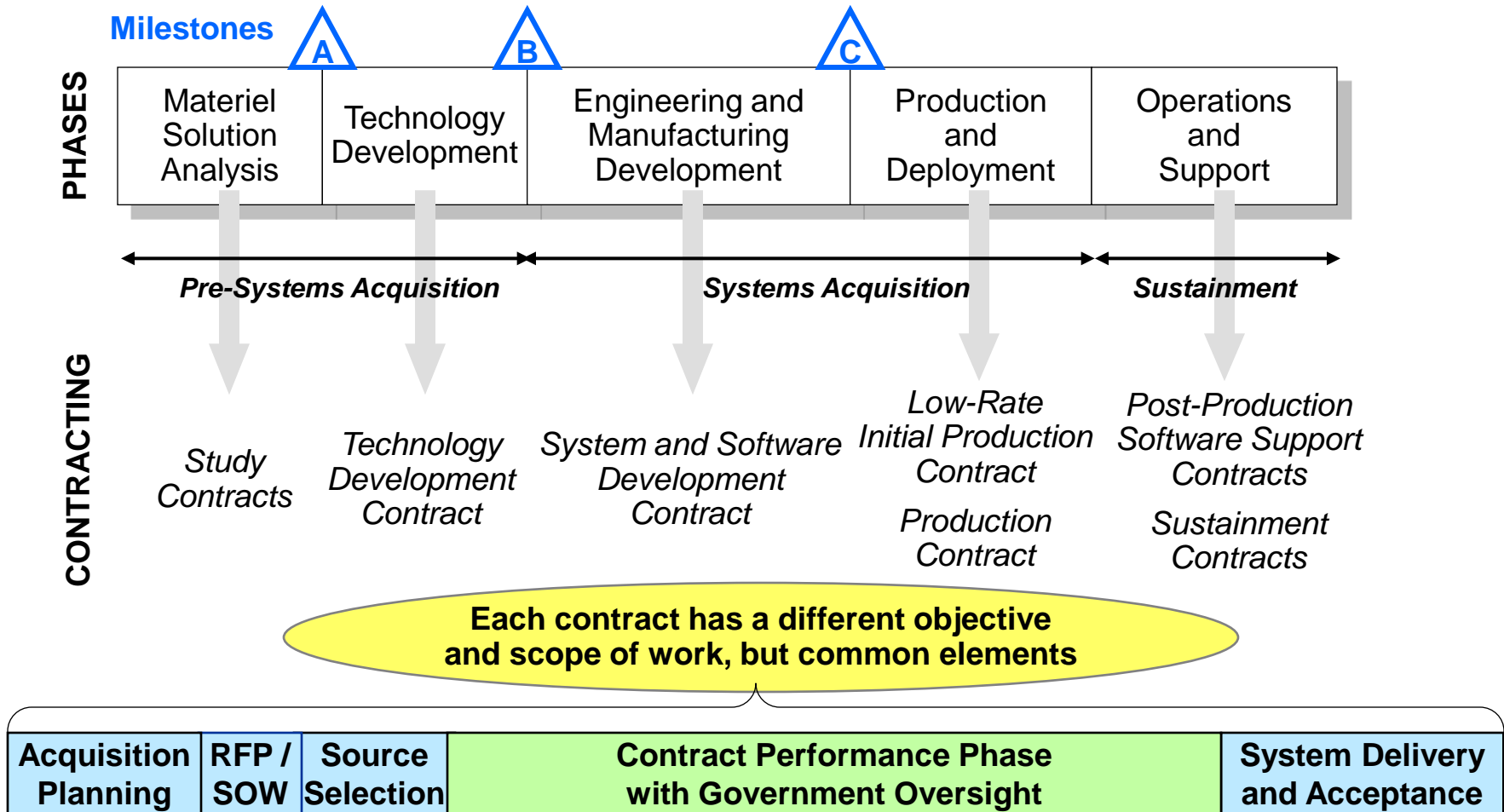
Acquisition is the process of obtaining products and services through contracting. Contracting includes purchasing, buying, commissioning, licensing, leasing, and procuring of designated supplies and services via a formal written agreement. [Bergey and Fisher]

**Contracting  
is the common  
denominator**



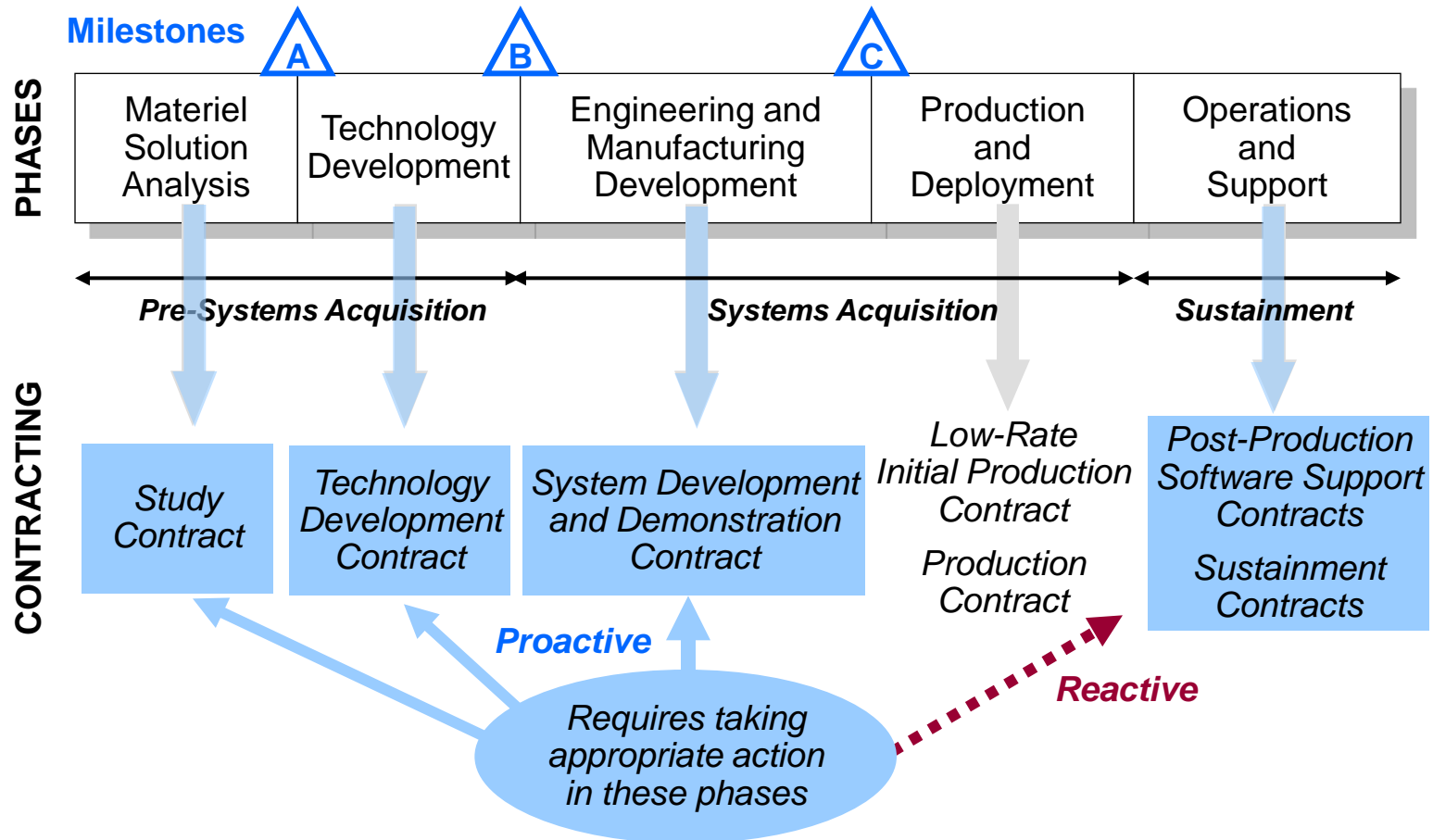


# Contractual View of DoD Acquisition Life Cycle



# Contractual View of DoD Acquisition Life Cycle

*– Adopting an Architecture-Centric Acquisition Approach –*

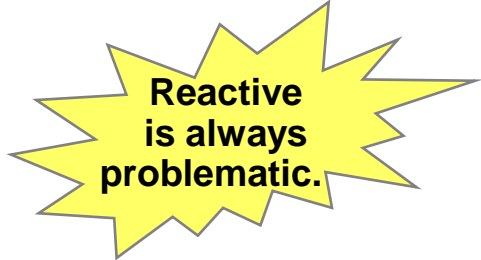




# Two Fundamental Ways Architecture-Centric Activities can be Incorporated in an Acquisition

## Reactive

Architecture-centric activities are *initiated opportunistically* and performed in situ under an existing contract at the request of the program manager.<sup>1</sup>



Reactive  
is always  
problematic.

## Proactive

Architecture-centric activities are *preplanned and integrated* up front in a request for proposal (RFP) for a system (or software) acquisition.

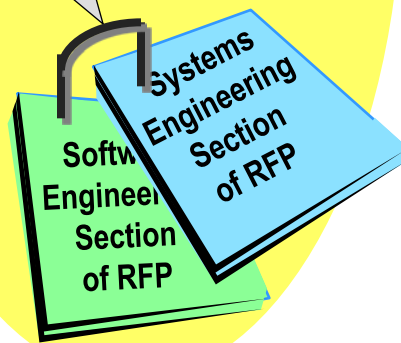
<sup>1</sup> Or at the request of a contractor under a negotiated agreement



# Integration of Systems and Software Engineering Aspects in an RFP

the way it's  
commonly  
done today

the “integrating  
element” of the  
system & software  
engineering  
aspects in the  
traditional RFP



**RFP Preparation**



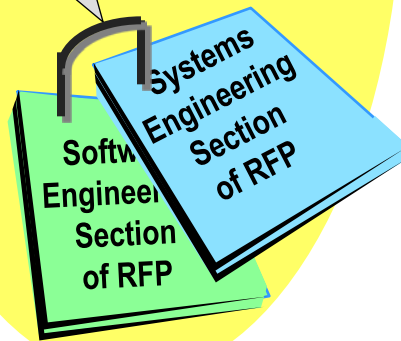
# Promoting System and Software Engineering Congruency in Acquisition

**FROM**  
the “staple”  
paradigm

**TO**

a synergy-driven  
paradigm

the “integrating  
element” of the  
system & software  
engineering  
aspects in the  
traditional RFP



**RFP Preparation**

## Definition of Synergy

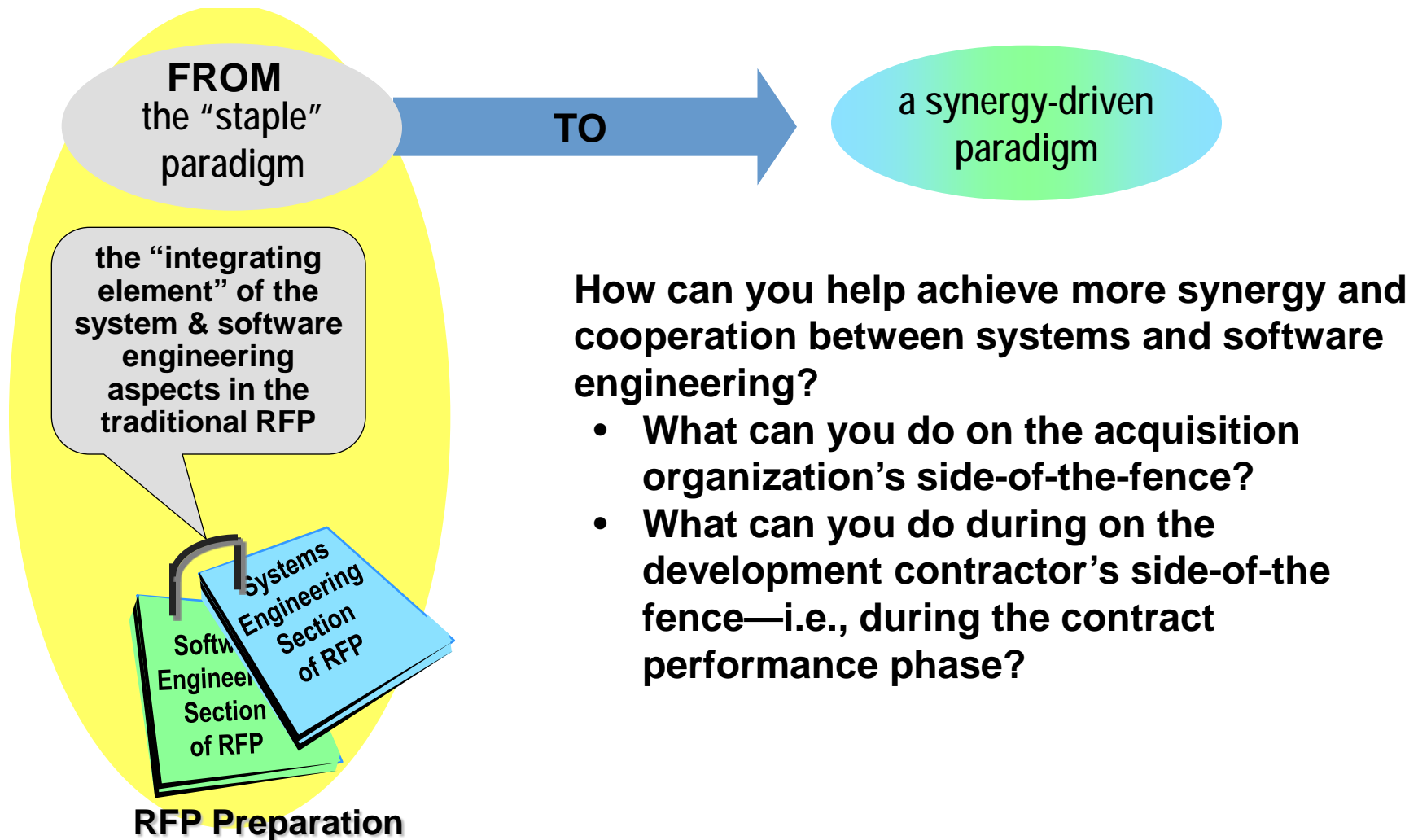
1. The interaction of two or more agents or forces so that their combined effect is greater than the sum of their individual effects.
2. Cooperative interaction among groups, especially among the acquired subsidiaries or merged parts of a corporation, that creates an enhanced combined effect.



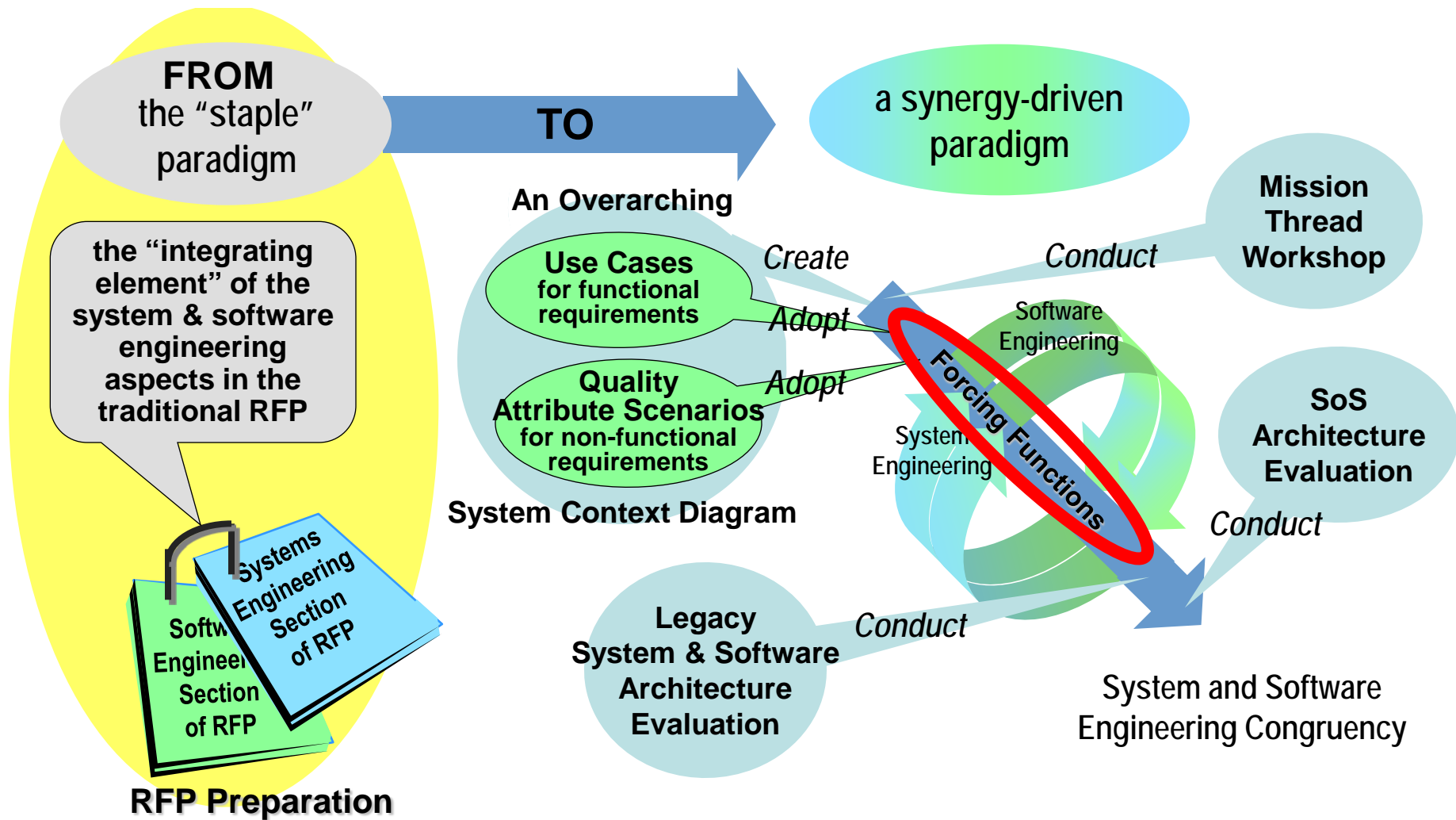
An Ask.com Service



# Promoting System and Software Engineering Congruency in Acquisition



# Promoting System and Software Engineering Congruency in Acquisition

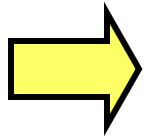


# DoD Acquirers and Software Architecture Practices

**What is** an architecture-centric acquisition approach?

**Why** should a DOD program adopt an architecture-centric acquisition approach?

**Where and when** is it appropriate in the DoD acquisition life cycle?



**How** can a DoD program adopt an architecture-centric approach and **what** does it involve?



# Key Elements of an Architecture-Centric Acquisition and Development Approach

## 1. Specifying a system's quality attributes

This involves conducting a Quality Attribute Workshop (QAW) with key stakeholders to elicit and capture<sup>1</sup> quality attribute scenarios (i.e., specify the non-functional requirements) so the architecture can be appropriately designed.

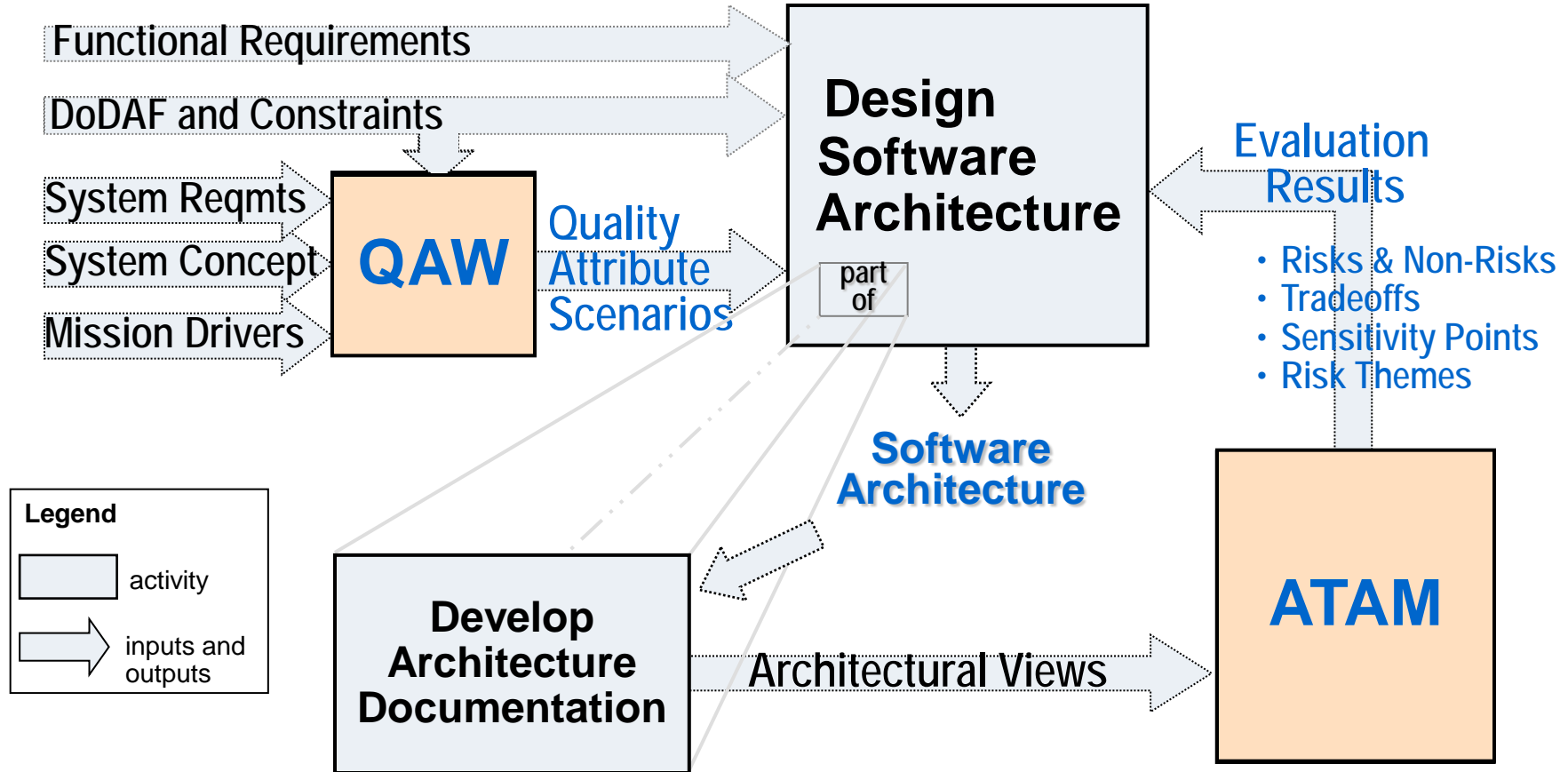
## 2. Evaluating the system and software architecture

This involves conducting an architecture evaluation (in collaboration with the system developer) using the SEI's Architecture Tradeoff and Analysis Method to identify and mitigate risks early in the system development cycle.

<sup>1</sup> To represent and record in a lasting form



# Architecture Centric Development

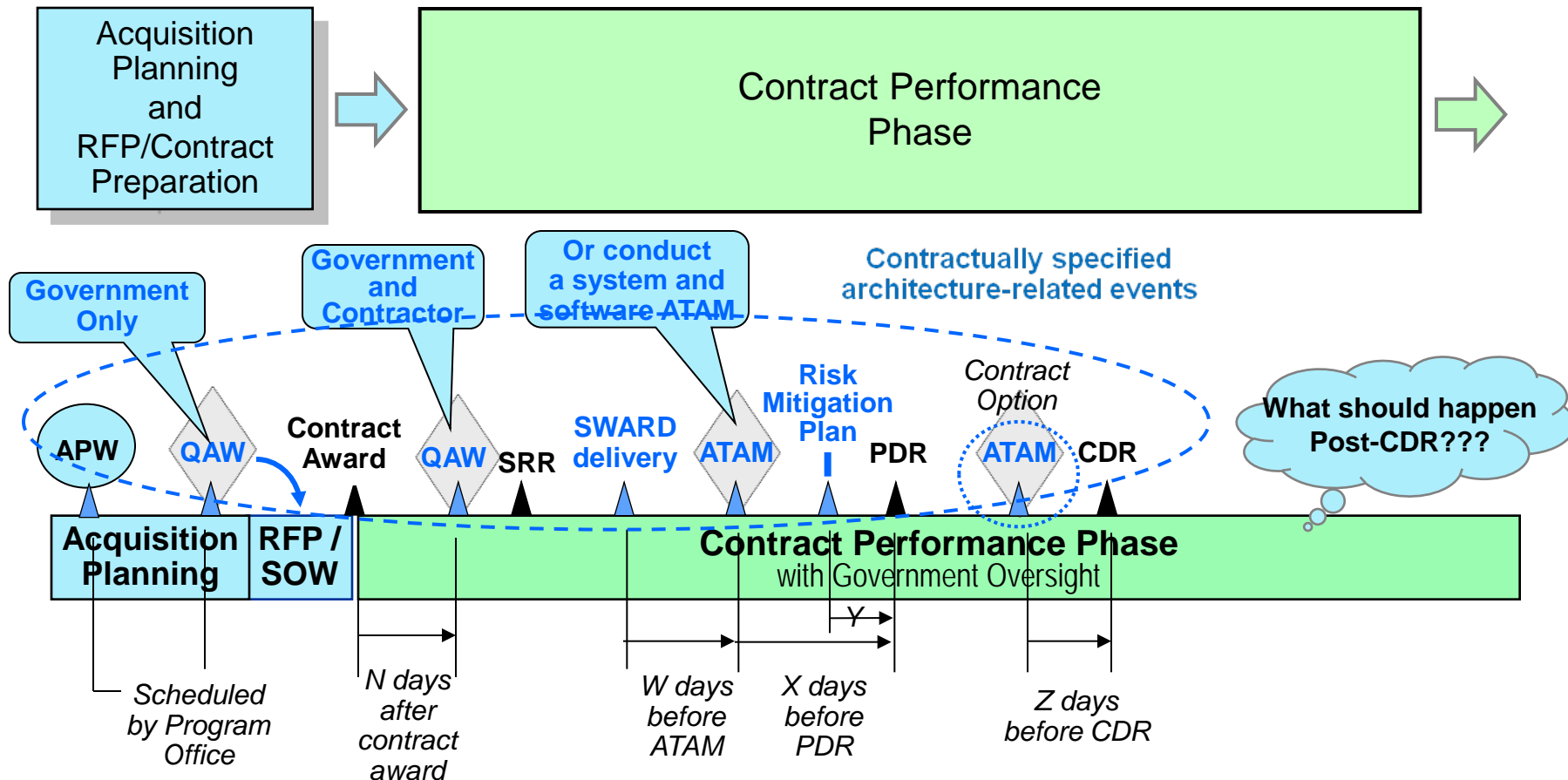


Such a “big picture” view of a contractor’s architecture-centric development approach would be described in its Software Development Plan (SDP).





# Elements of an Architecture-Centric Acquisition Approach



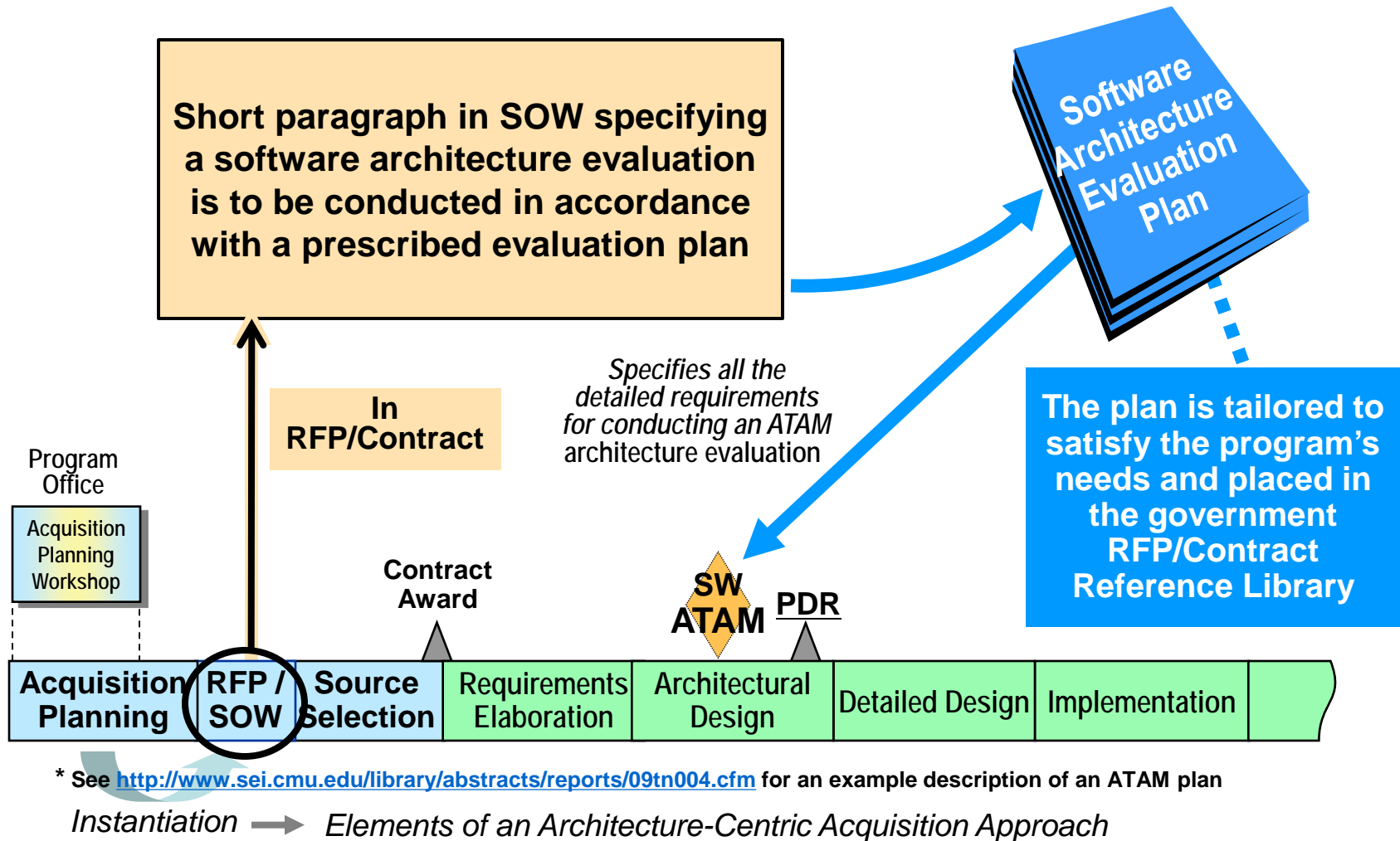
**Legend**

- APW – Acquisition Planning Workshop
- PDR – Preliminary Design Review
- SRR – System Requirements Review

- SWARD – Software Architecture Description Document
- CDR – Critical Design Review



# Model for Incorporating Software Architecture Evaluation in an RFP/Contract



# Ensuring a Coherent Approach is Adopted

Document	Type of Information to Be Included (Relative to Conducting an Architecture Evaluation)
<b>SEMP</b>	Describe: (1) how the architecture evaluation is integrated into the system engineering management plan in relation to the program milestones, (2) how the system's quality attribute requirements (i.e., nonfunctional requirements) that drive the architectural design will be specified and managed, and (3) how the software architecture will be documented.
<b>TEMP</b>	Describe the role of architecture evaluation in the test and evaluation management plan and when the evaluation will be scheduled in relation to the program milestones.
<b>SEP</b>	Describe: (1) how the architecture evaluation is integrated into the system engineering plan in relation to the system engineering milestones, (2) how the system's quality attribute requirements (i.e., nonfunctional requirements) that drive the architectural design will be specified and managed, and (3) how the software architecture will be documented.
<b>SDP</b>	Describe how the software architecture evaluation fits into the overall software development approach including how identified risks (and risk themes) will be analyzed and mitigated.
<b>STP</b>	Describe the role of architecture evaluation in the software test plan and when the evaluation will be scheduled in relation to software testing milestones.
<b>RMP</b>	Describe how risks (and risk themes) emanating from the architecture evaluation will be integrated with the program's risk management system and subsequently managed (i.e., identified, tracked, and mitigated).



# Example: Architecture Aspects You May Want the Offerror to Describe in their Technical Proposal

## Section L – Instructions to Offerrors

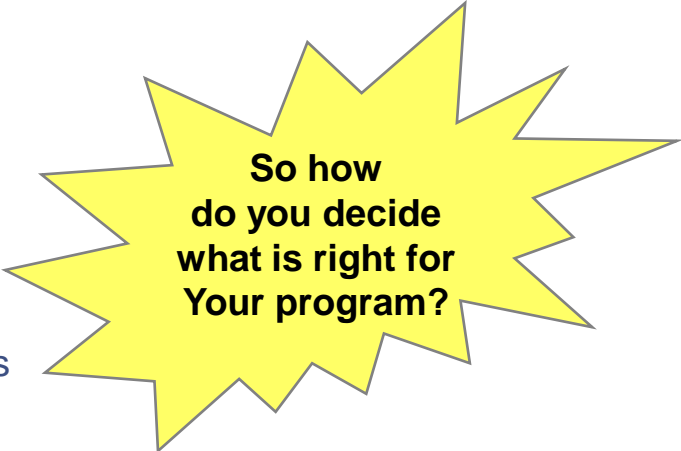
1. Describe how quality attribute scenarios resulting from the QAW will be integrated into the requirements baseline and managed from that point forward.
2. Describe how architecture risks and risk themes discovered during the ATAM evaluation will be prioritized and mitigated.
3. Describe how proposed software modifications (including architectural changes) that occur during the system life cycle will be managed.
4. Describe how compliance of the software implementation with the approved software architecture baseline will be enforced throughout the life cycle.
5. Describe what kind of software architecture metrics will be collected and reported to the government during the contract performance phase.



# Examples of Architecture-Centric Practices that Can Be Incorporated in an RFP/Contract

Architecture-centric activities, deliverables and measures that can easily be incorporated into an RFP today for a system acquisition include:

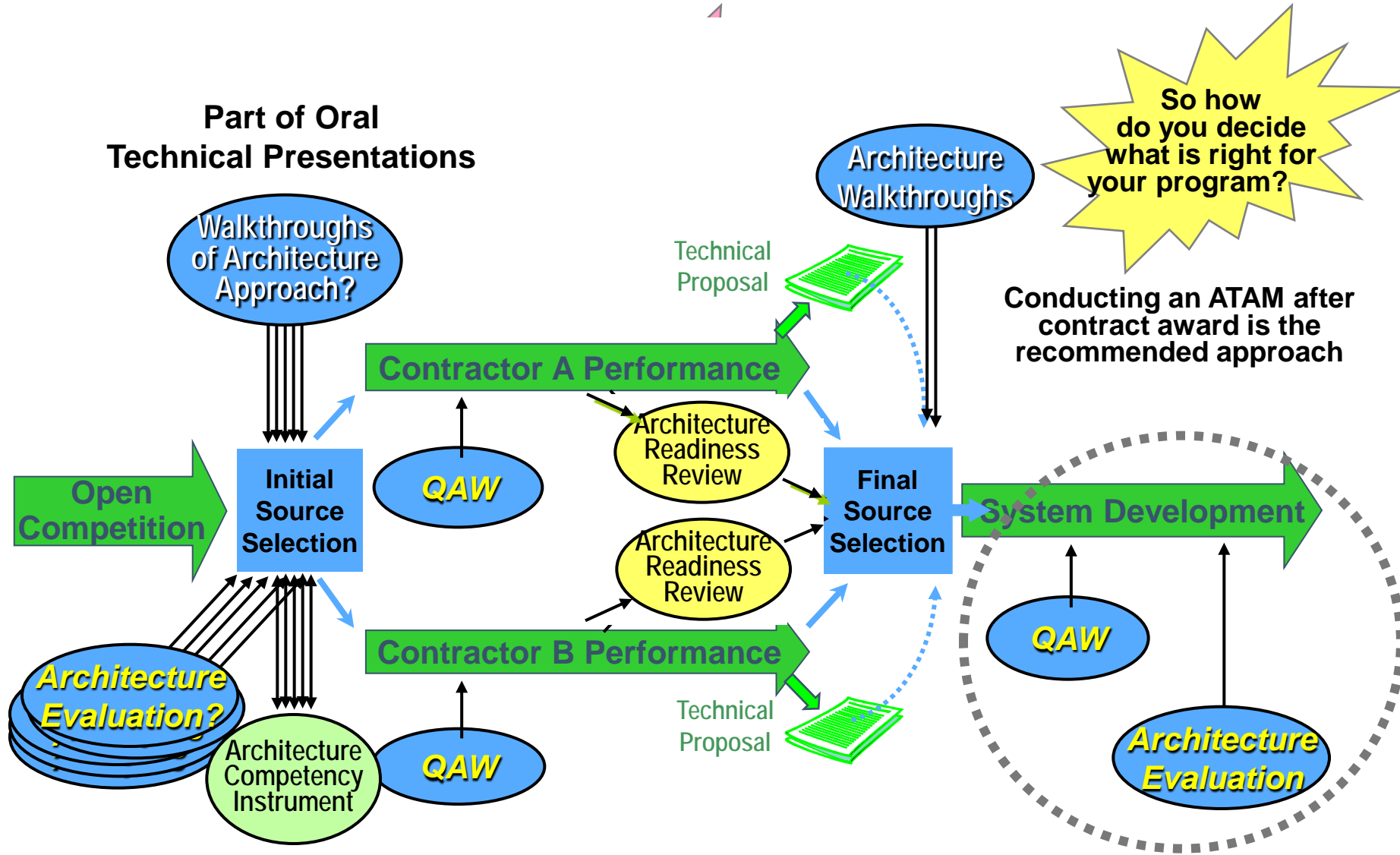
- **Initial specification of quality attribute requirements**
- **Quality Attribute Workshop (QAW)** to collaboratively
  - Validate business and mission drivers
  - Elicit quality attributes (system and software)
  - Refine a set of quality attribute scenarios
- **Architecture competency instrument**
  - Assessment of offeror's architecture experience/expertise for use in source selection
- **Architecture design and evolution guidance**
  - Quality attribute-driven architectural design
- **Software architecture description**
  - Include as part of contractual deliverables
- **Architecture Readiness Review (ARR)**
- **Software architecture evaluation**
  - Specify collaborative evaluation based on ATAM
  - Require evaluation report identifying risks and risk themes
- **Risk mitigation "monitoring instrument"**
  - Monitor the risk mitigation activities and report on progress
- **Cost benefit change analysis**
  - Prioritization of architecture risk mitigation activities based on cost benefit



**So how  
do you decide  
what is right for  
Your program?**



# Common Architecture Evaluation Scenarios



# Conducting an Acquisition Planning Workshop



## Why hold a workshop?

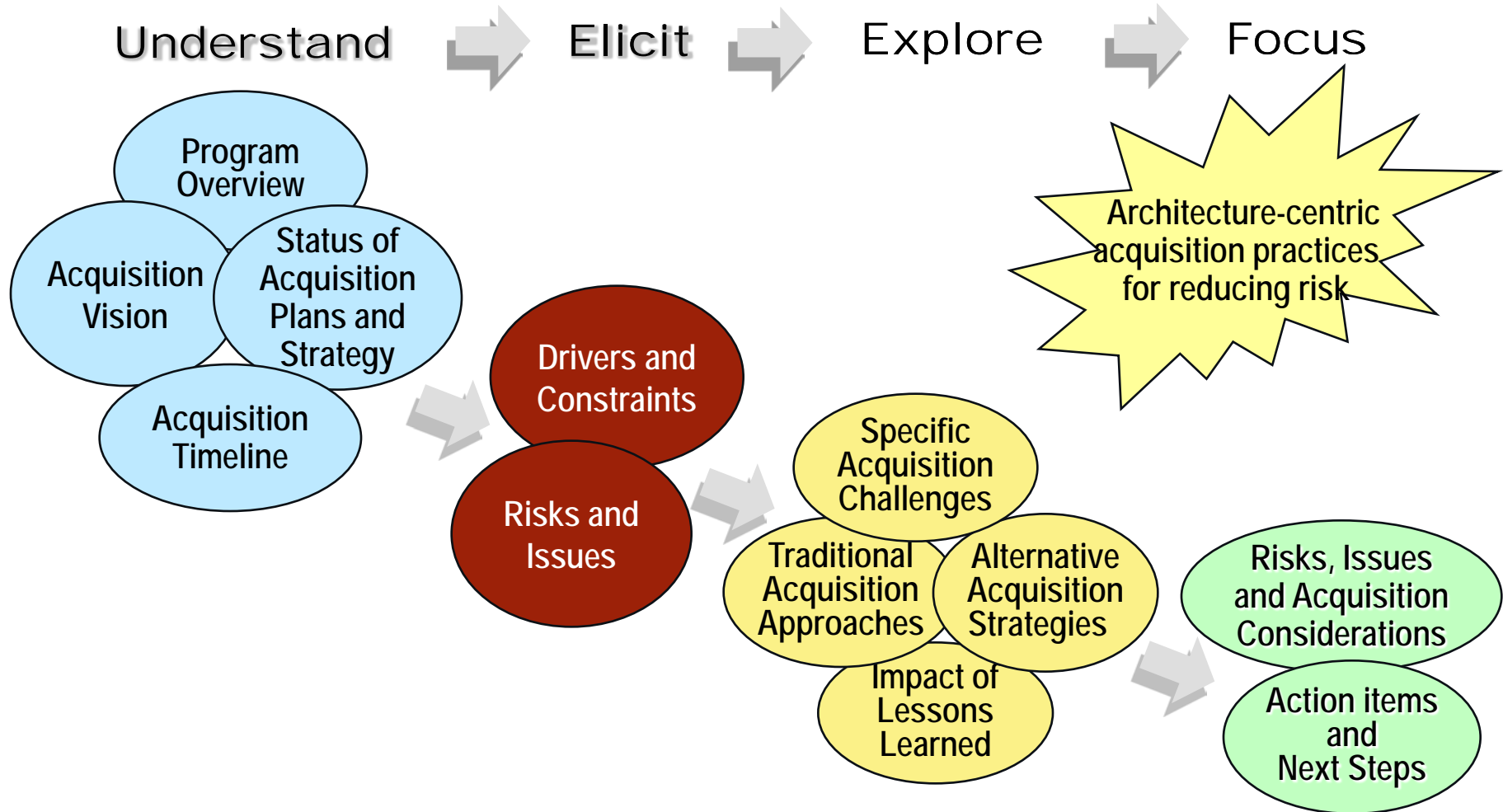
1. To be **proactive** and provide upfront assistance during the **acquisition planning and RFP preparation phase** when it can make a difference.
2. To provide a **structured forum** for key acquisition stakeholders to **understand** the program's acquisition approach and current status, **and explore** potential ways for **reducing software acquisition risk** via a facilitated technical interchange

## Outputs

1. Common understanding of the acquisition challenges, risks, and key issues
2. A list of actions for going forward with acquisition planning



# Overview of Acquisition Planning Workshop





# How Acquisition Programs Can Leverage an Architecture-Centric Approach to Reduce Risk

## Realize that Architecture is Key

- Embodies the early design decisions that are the most difficult to get right
- Provides level-of-abstraction best aligned with program responsibilities

## Focus on quality attributes

- Allows stakeholders to discuss, clarify, and prioritize non-functional requirements that are often problematic

## Acquire Comprehensive Architecture Documentation

- Provides the means to analyze the software design and guide development

## Evaluate the System and Software Architecture

- Promotes coordination between system and software engineering

## Focus on Risk Management

- Risk identification and mitigation

## Arrange for Training

- Educate both program office and contract personnel

## Conduct an Acquisition Planning Workshop

- Be proactive and ensure the right stuff gets in the RFP/contract



# Summary -1

The reasons for adopting an architecture-centric acquisition approach are:

- Product quality cannot be tested in, it must be designed in.
- Quality processes greatly influence product quality but are not sufficient to guarantee a quality product.
- Architecture is the earliest design artifact that represents the indispensable first step towards a solution.
- Architecture not only structures the system, but structures the process of building the system; the discrete parts representing separable work assignments.
- Architecture largely determines the quality attributes (performance, modifiability, security, etc.) of the resulting system and is amenable to evaluation.



# Summary -2

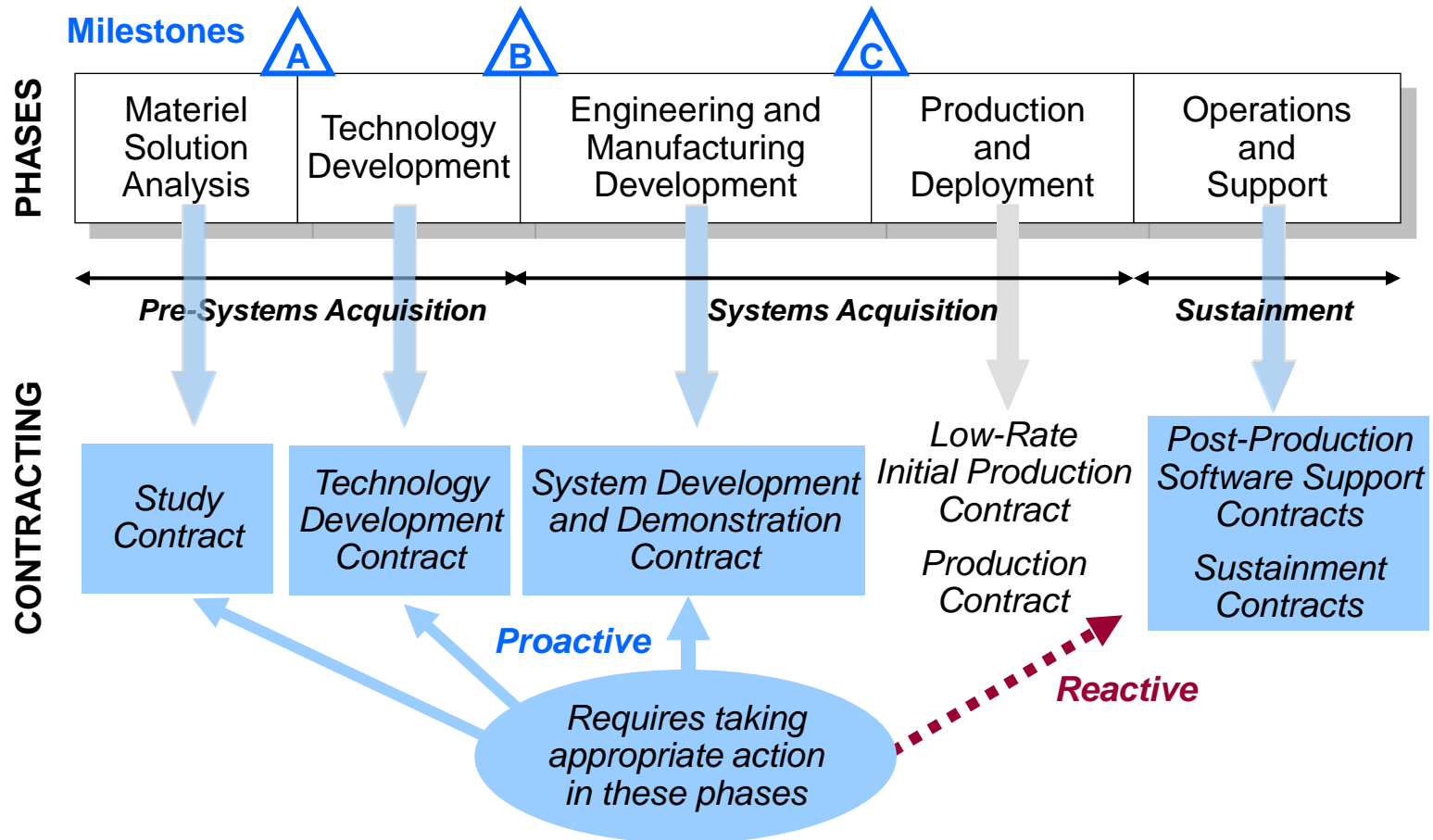
Moreover, an architecture-centric acquisition approach

- enables a program office to perform its technical oversight and technical monitoring function with greater effectiveness
  - commensurate with a program office's responsibilities, limited resources, time available, and key contractual events
  - provides early insight into critical requirements and design decisions that drive the entire development effort
  - provides a proven and effective means for discovering software design risks and risk themes
  - enables risks to be mitigated early and cost effectively and avoids costly rework downstream
- results in the delivery of a more capable and higher quality product to the warfighter

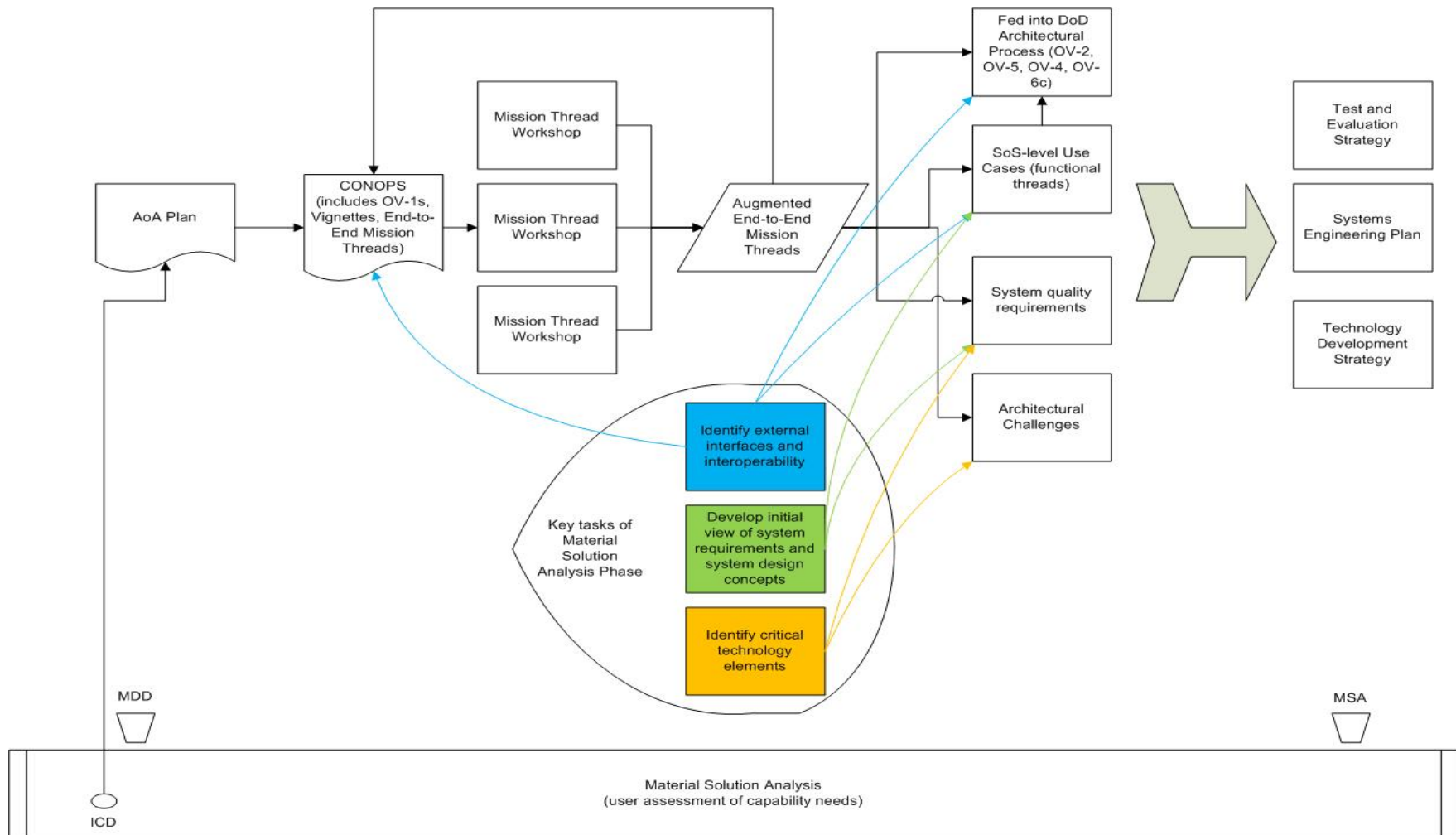


# Contractual View of DoD Acquisition Life Cycle

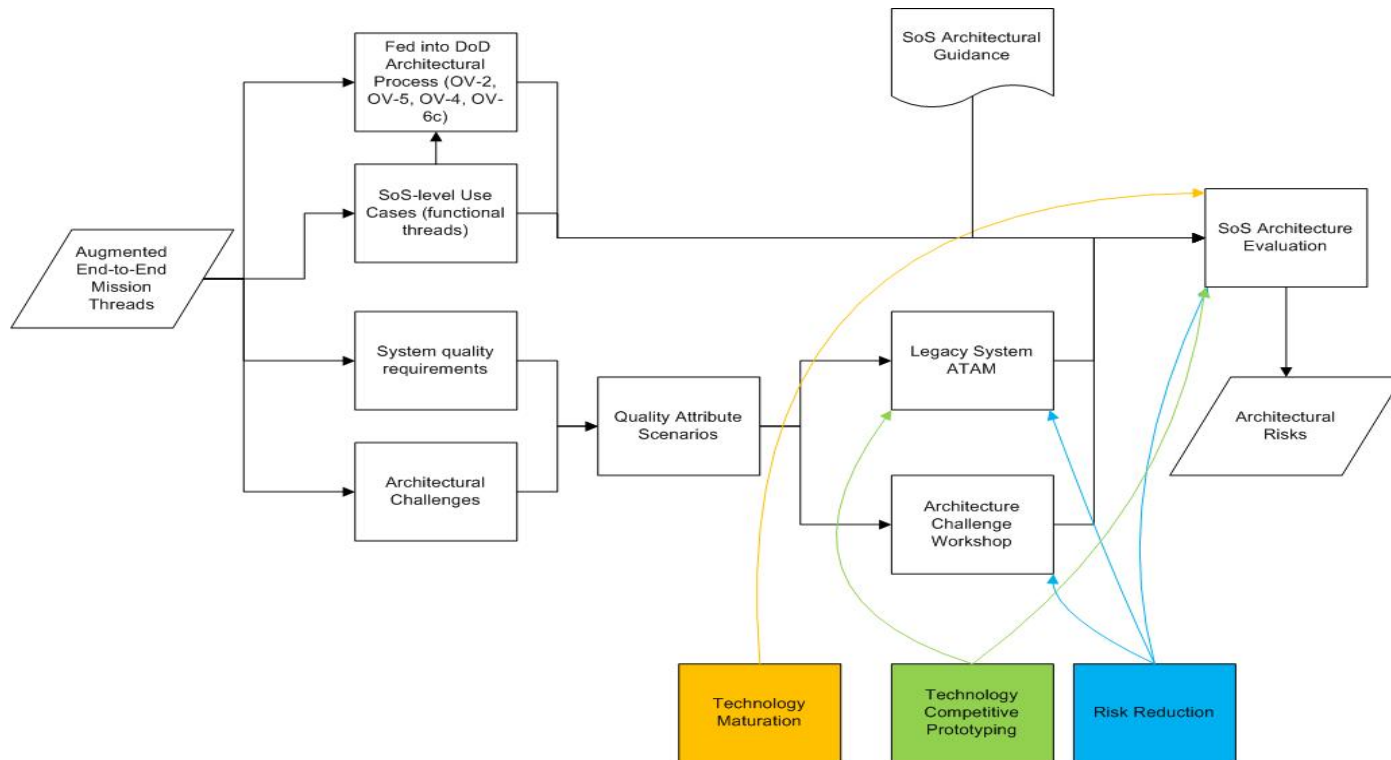
*– Adopting an Architecture-Centric Acquisition Approach –*



# Material Solutions Analysis Phase



# Technology Development Phase



# Value at Various Stages - 1

Working early with the Program Office to develop the proper architecture-centric acquisition strategy and associated language for proposals, contracts, etc. will

- drive the contractors to do the right thing architecturally early
- provide visibility to the program office into the architecture's goodness
- identify architectural risks early

This is the biggest point of leverage within DoD programs. We have demonstrated its effectiveness on DoD programs in software architecture. Our many pilots indicate that this is true for SoS and system architecture as well.



# Value at Various Stages - 2

MTW - Early elicitation of SoS quality attribute needs, architectural challenges, mission and capability challenges, system use cases.

- Stakeholder-elicited quality attribute information available to the SoS architecture developers (and integrators and testers); also used to inform the system and software architecture development/acquisition activities.
- Challenges are identified early in the life cycle, to prevent them from becoming risks later.

Architecture Evaluation - Early identification of SoS architecture risks and problematic constituent system and software architectures.

- The architects, along with the program office, can identify, prioritize, and mitigate risks early in the life cycle, prior to integration.
- Addressing the risks prior to integration will reduce integration and operational risks.





# Contact Information

## Michael Gagliardi

Research, Technology, and Systems  
Solutions Program

Telephone: 412-268-7738

Email: [mjg@sei.cmu.edu](mailto:mjg@sei.cmu.edu)

## John Bergey

Research, Technology, and Systems  
Solutions Program

Telephone: 215-348-0530

Email: [jkb@sei.cmu.edu](mailto:jkb@sei.cmu.edu)

## Linda Northrop

Director: Research, Technology, and Systems  
Solutions Program

Telephone: 412-268-7638

Email: [lmn@sei.cmu.edu](mailto:lmn@sei.cmu.edu)

## U.S. Mail:

Software Engineering Institute  
Carnegie Mellon University  
4500 Fifth Avenue  
Pittsburgh, PA 15213-3890

## World Wide Web:

<http://www.sei.cmu.edu/productlines>

SEI Fax: 412-268-5758



## NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.





**Software Engineering Institute**

**Carnegie Mellon**



**Software Engineering Institute**

**Carnegie Mellon**

**SATURN 2010 Tutorial**  
**Gagliardi, Bergey**  
© 2010 Carnegie Mellon University